

DEPARTMENT OF MECHANICAL ENGINEERING AND MECHANICS  
SCHOOL OF ENGINEERING  
OLD DOMINION UNIVERSITY  
NORFOLK, VIRGINIA 23508

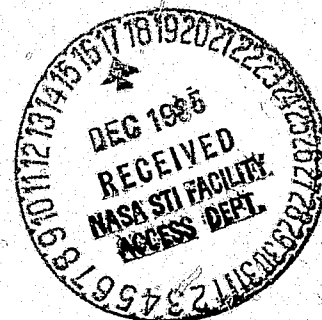
THREE-DIMENSIONAL ELASTIC-PLASTIC FINITE-ELEMENT  
ANALYSIS OF FATIGUE CRACK PROPAGATION

By

R. G. Chermahini, Research Associate

and

G. L. Gogia, Principal Investigator



Final Report

For the period June 1, 1985 to November 1, 1985

Prepared for  
National Aeronautics and Space Administration  
Langley Research Center  
Hampton, VA 23665

Under  
Research Grant NAG-1-529  
Dr. James C. Newman, Jr., Technical Monitor  
MD-Fatigue & Fracture Branch

(NASA-CR-176415) THREE-DIMENSIONAL  
ELASTIC-PLASTIC FINITE-ELEMENT ANALYSIS OF  
FATIGUE CRACK PROPAGATION Final Report, 1  
Jun. - 1 Nov. 1985 (Old Dominion Univ.,  
Norfolk, Va.) 60 p HC A04/MF A01 CSCL 20K G3/39

N86-14687

Unclas

04979

November 1985



DEPARTMENT OF MECHANICAL ENGINEERING AND MECHANICS  
SCHOOL OF ENGINEERING  
OLD DOMINION UNIVERSITY  
NORFOLK, VIRGINIA 23508

THREE-DIMENSIONAL ELASTIC-PLASTIC FINITE-ELEMENT  
ANALYSIS OF FATIGUE CRACK PROPAGATION

By

R. G. Chermahini

and

G. L. Goglia, Principal Investigator

Final Report

For the period June 1, 1985 to November 1, 1985

Prepared for

National Aeronautics and Space Administration  
Langley Research Center  
Hampton, VA 23665

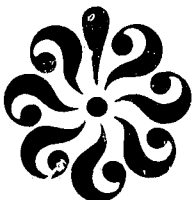
Under

Research Grant NAG-1-529

Dr. James C. Newman, Jr., Technical Monitor  
MD-Fatigue & Fracture Branch

Submitted by the

Old Dominion University Research Foundation  
P.O. Box 6369  
Norfolk, Virginia 23508



November 1985

## TABLE OF CONTENTS

	<u>Page</u>
INTRODUCTION.....	1
APPENDIX A.....	4
Finite Element Formulation.....	4
(a) Elastic Analysis.....	4
Displacement Functions.....	4
Element Strain.....	9
Element Stress.....	11
Element Equations.....	12
(b) Elastic-plastic Analysis.....	12
Yield Criterion.....	14
APPENDIX B.....	18
Description of the Finite-Element Computer Program.....	18
REFERENCES.....	

## LIST OF FIGURES

<u>Figure</u>	<u>Page</u>
1 Cyclic stress-strain curve for an elastic-perfectly plastic material.....	3
2 Arbitrary hexahedron.....	4
3 Linear isoparametric cubic element.....	5
4 Center-Crack Panel subjected to uniform stress.....	19

# THREE-DIMENSIONAL ELASTIC-PLASTIC FINITE-ELEMENT ANALYSIS OF FATIGUE CRACK PROPAGATION

BY

R. G. Chermahini<sup>1</sup> and G. L. Goglia<sup>2</sup>

## INTRODUCTION

Fatigue cracks have been a major problem in designing structures subjected to cyclic loading. Cracks frequently occur in structures such as aircraft and spacecraft. The inspection intervals of many aircraft structures are based on crack-propagation lives. Therefore, improved prediction of propagation lives under flight-load conditions (variable-amplitude loading) are needed to provide more realistic design criteria for these structures.

The main thrust of this study was to develop a three-dimensional, non-linear, elastic-plastic, finite element program capable of extending a crack and changing boundary conditions for the model under consideration. The finite-element model is composed of 8-noded (linear-strain) isoparametric elements. In the analysis, the material is assumed to be elastic-perfectly plastic. The cycle stress-strain curve for the material is shown in Fig. 1. Zienkiewicz's "initial-stress" method, von Mises's yield criterion, and Drucker's normality condition under small-strain assumptions are used to account for plasticity. The three-dimensional analysis is capable of extending the crack and changing boundary conditions under cyclic loading. Initially, the crack is assumed to grow as a straight-through crack.

---

<sup>1</sup>Research Associate, Department of Mechanical Engineering & Mechanics, Old Dominion University, Norfolk, VA 23508.

<sup>2</sup>Eminent Professor, Department of Mechanical Engineering & Mechanics, Old Dominion University, Norfolk, VA 23508.

Using a three-dimensional nonlinear computer program on a cyber-nos system was impossible due to its limited storage capacity. To avoid this problem, the next alternative was to utilize a VPS-32 machine with unlimited storage capacity. Using the scalar version of the program on the VPS-32 was costly due to the plasticity part of the program. Therefore, in order to reduce the cost of the computations, the three-dimensional computer program was vectorized.

The finite-element formulation of the program using an 8-noded linear isoparametric cubic element is listed in Appendix A. The description of the nonlinear program is attached in Appendix B. A list of the program is shown in Appendix C.

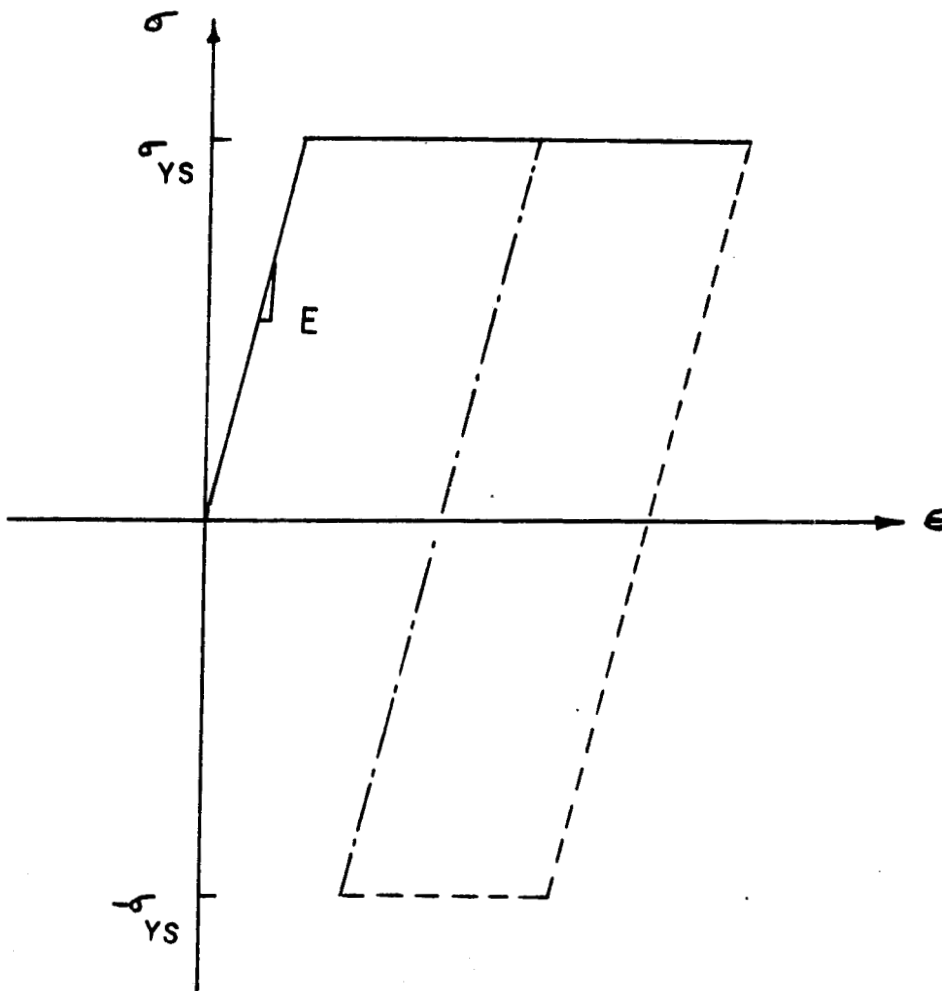


Figure 1. CYCLIC STRESS-STRAIN CURVE FOR AN ELASTIC-PERFECTLY PLASTIC MATERIAL

## APPENDIX A

### FINITE-ELEMENT FORMULATION

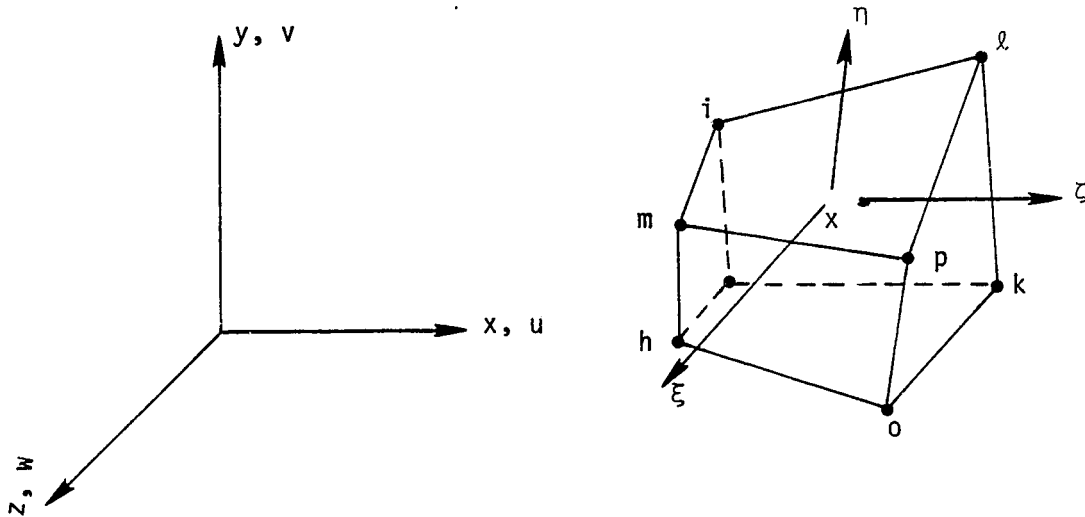


Figure 2. Arbitrary hexahedron.

#### (a) Elastic Analysis

The basic concept of the finite-element method is that any continuous quantity can be approximated by a discrete model composed of a set of piecewise continuous functions defined over a finite number of subdomains (1).

An isoparametric 8-noded cubic element (Fig. 2.) was utilized in the formulation of the elastic-plastic structure into the nonlinear computer program. The following section describes cubic element.

Displacement Functions. The displacement function (2) for any point in the cubic element is defined as:

$$\begin{aligned}
u(x,y,z) &= a_1 + a_2x + a_3y + a_4z + a_5xy + a_6yz + a_7xz + a_8xyz \\
v(x,y,z) &= b_1 + b_2x + b_3y + b_4z + b_5xy + b_6yz + b_7xz + b_8xyz \\
w(x,y,z) &= c_1 + c_2x + c_3y + c_4z + c_5xy + c_6yz + c_7xz + c_8xyz
\end{aligned} \tag{A(1)}$$

where  $u$ ,  $v$  and  $w$  are displacement in the  $x$ ,  $y$  and  $z$  directions, respectively. The constant coefficients are determined by imposing the nodal coordinates of each cubic element into equations A (1). The above displacement function can be applied to the cubic element as long as the sides of the cubic element are defined by planes parallel to the coordinate planes. However, for the elements whose sides are skewed, the above displacement function no longer is applicable. Therefore, in order to avoid this restriction, an 8-noded linear isoparametric cubic element is employed (Fig. 2.).

The original cube can be mapped on to a cube of  $2 \times 2 \times 3$  unit (2) in the  $\zeta$ ,  $\eta$ ,  $\xi$  space by the transformation

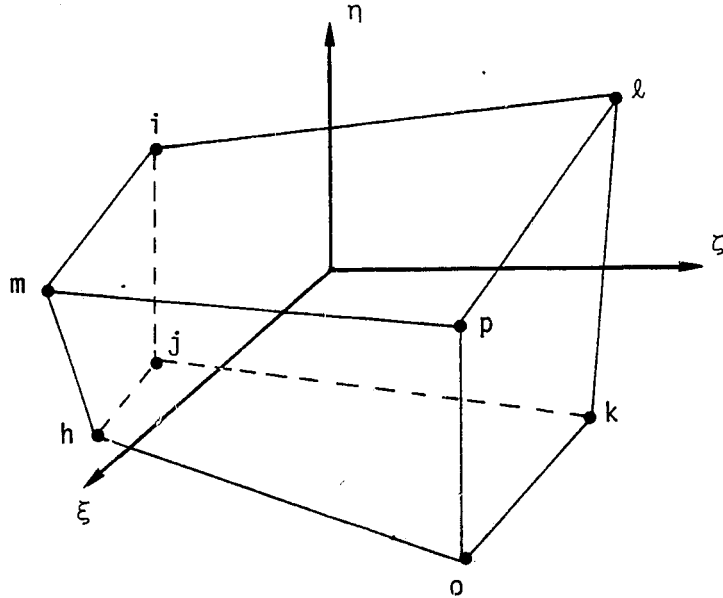
$$\begin{aligned}
x &= a_1 + a_2\xi + a_3\eta + a_4\zeta + a_5\xi\eta + a_6\eta\xi + a_7\zeta\xi + a_8\zeta\eta\xi \\
y &= b_1 + b_2\xi + b_3\eta + b_4\zeta + b_5\xi\eta + b_6\eta\xi + b_7\zeta\xi + b_8\zeta\eta\xi \\
z &= c_1 + c_2\zeta + c_3\eta + c_4\xi + c_5\xi\eta + c_6\eta\xi + c_7\zeta\xi + c_8\zeta\eta\xi
\end{aligned} \tag{A(2)}$$

The values of the coefficients in equation A(2) depend on the nodal coordinates of each cubic element and are different for different elements. The transformation is defined by polynomials in  $\zeta$ ,  $\eta$  and  $\xi$  which is continuous within the element, the continuum confined within an element in  $x$ ,  $y$  and  $z$  coordinates is mapped on to a continuum within the  $2 \times 2 \times 2$  cube in  $\zeta$ ,  $\eta$  and  $\xi$  coordinates. It remains to be shown that the transformation is continuous across two adjoined elements, that a common surface between



two adjoined elements in the  $x, y, z$  space will transform into a common surface of two adjoined cubes in  $\zeta, \eta, \xi$  space.

If we assign the following values of the parameters  $\zeta, \eta, \xi$  to the faces of distorted elements shown in (Fig. 3.) one yields:



<u>Face</u>	<u>Coordinate value</u>
pokl	$\zeta = 1$
mnji	$\zeta = -1$
impl	$\eta = 1$
jnok	$\eta = -1$
mnop	$\xi = 1$
ijkl	$\xi = -1$

Fig. 3. Linear isoparametric cubic element.

Therefore, the nodal points i, j, k, l and m, n, o, p will have the following coordinates in the  $\zeta, \eta, \xi$ :

nodal point	coordinates		
i	$\zeta_i = -1$	$\eta_i = 1$	$\xi_i = -1$
j	$\zeta_j = -1$	$\eta_j = -1$	$\xi_j = -1$
k	$\zeta_k = 1$	$\eta_k = -1$	$\xi_k = -1$
l	$\zeta_l = 1$	$\eta_l = 1$	$\xi_l = -1$
m	$\zeta_m = -1$	$\eta_m = 1$	$\xi_m = 1$
n	$\zeta_n = -1$	$\eta_n = -1$	$\xi_n = 1$
o	$\zeta_o = 1$	$\eta_o = -1$	$\xi_o = 1$
p	$\zeta_p = 1$	$\eta_p = 1$	$\xi_p = 1$

Now the displacements (u, v, w) in the x, y, z directions can be written as:

$$\begin{aligned}
 u &= \alpha_1 + \alpha_2 \zeta + \alpha_3 \eta + \alpha_4 \xi + \alpha_5 \xi \eta + \alpha_6 \eta \xi + \alpha_7 \zeta \xi + \alpha_8 \zeta \eta \xi \\
 v &= \beta_1 + \beta_2 \xi + \beta_3 \eta + \beta_4 \zeta + \beta_5 \zeta \eta + \beta_6 \eta \xi + \beta_7 \zeta \xi + \beta_8 \zeta \eta \xi \\
 w &= \gamma_1 + \gamma_2 \zeta + \gamma_3 \eta + \gamma_4 \xi + \gamma_5 \xi \eta + \gamma_6 \eta \xi + \gamma_7 \zeta \xi + \gamma_8 \zeta \eta \xi
 \end{aligned} \tag{A(3)}$$

which are continuous (2) within the elements as well as across the surfaces common to any two adjoined elements. Consider the term u in equations A(3), denote by {a} and {U} the vectors for the  $\alpha$ 's and  $u_i$  nodal displacements of all the nodal points of the element. Inserting the values of  $u_i$ ,  $\zeta_i$ ,  $\eta_i$  and  $\xi_i$  for the various nodal points, we obtain eight equations corresponding to the first equation of A(3) which can be written as

$$\{U\} = [A_1] \{a\} \tag{A(4)}$$

Let's define  $[\alpha_1] = [A_1]$ , and thus have  $\{a\} = [\alpha_1] \{u\}$ . Now the displacement functions for the distorted element can be written as:

$$\begin{aligned} u &= [S] [\alpha_1] \{u\} \\ v &= [S] [\alpha_1] \{v\} \\ w &= [S] [\alpha_1] \{w\} \end{aligned} \quad A(5)$$

where  $[S]$  is defined as:

$$[S] = [1 \ \zeta \ \eta \ \xi \ \zeta\eta \ \eta\xi \ \zeta\xi \ \zeta\eta\xi] \quad A(6)$$

The shape functions for the isoparametric 8-noded element can be determined (1) from the product of  $[S]$  and  $[\alpha_1]$  matrices.

$$N_i = \frac{1}{8} (1+\zeta_i) (1+\eta_i) (1+\xi_i) \quad A(7)$$

where  $\zeta_i, \eta_i, \xi_i = \pm 1$  and  $i = 1, 2, \dots, 8$ .

The  $x$ ,  $y$ , and  $z$  coordinates at any point in the element, can be expressed in terms of shape functions  $N_i$ :

$$\begin{aligned} x &= \sum_{i=1}^8 N_i X_i \\ y &= \sum_{i=1}^8 N_i Y_i \\ z &= \sum_{i=1}^8 N_i Z_i \end{aligned} \quad A(8)$$

or

$$\begin{Bmatrix} x \\ y \\ z \end{Bmatrix} = \begin{bmatrix} [N] & [0] & [0] \\ [0] & [N] & [0] \\ [0] & [0] & [N] \end{bmatrix} \begin{Bmatrix} \{x_n\} \\ \{y_n\} \\ \{z_n\} \end{Bmatrix}$$

where  $\{x_n\}^T = [x_1 \ x_2 \ \dots \ x_8]$ ,  $\{y_n\}^T = [y_1 \ y_2 \ \dots \ y_8]$

and  $\{z_n\}^T = [z_1 \ z_2 \ \dots \ z_8]$ .

Element Strain: The elastic strain at any point within the element is given by [3]

$$\{\epsilon\} = \begin{Bmatrix} \epsilon_x \\ \epsilon_y \\ \epsilon_z \\ \gamma_{xy} \\ \gamma_{yz} \\ \gamma_{zx} \end{Bmatrix} = \begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial v}{\partial y} \\ \frac{\partial w}{\partial z} \\ \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \\ \frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \\ \frac{\partial w}{\partial x} + \frac{\partial u}{\partial z} \end{Bmatrix} = \begin{bmatrix} [B_1] & [B_2] & \dots & [B_8] \end{bmatrix} \{u\} = [B] \{u\} \quad A(9)$$

where the matrix  $[B]$  is defined as:

$$[B_i] = \begin{bmatrix} \frac{\partial N_i}{\partial x} & 0 & 0 \\ 0 & \frac{\partial N_i}{\partial y} & 0 \\ 0 & 0 & \frac{\partial N_i}{\partial y} \\ \frac{\partial N_i}{\partial y} & \frac{\partial N_i}{\partial x} & 0 \\ 0 & \frac{\partial N_i}{\partial z} & \frac{\partial N_i}{\partial y} \\ \frac{\partial N_i}{\partial z} & 0 & \frac{\partial N_i}{\partial x} \end{bmatrix} \quad A(10)$$

The transformation relationship between local and global coordinates is given by:

$$\begin{Bmatrix} \frac{\partial N_i}{\partial x} \\ \frac{\partial N_i}{\partial y} \\ \frac{\partial N_i}{\partial z} \end{Bmatrix} = [J]^{-1} \begin{Bmatrix} \frac{\partial N_i}{\partial \xi} \\ \frac{\partial N_i}{\partial \eta} \\ \frac{\partial N_i}{\partial \xi} \end{Bmatrix} \quad A(11)$$

where  $[J]$  is the Jacobian matrix and it is defined as:

$$[J] = \begin{bmatrix} \frac{\partial \{N\}^T}{\partial \xi} \\ \frac{\partial \{N\}^T}{\partial \eta} \\ \frac{\partial \{N\}^T}{\partial \xi} \end{bmatrix} \begin{Bmatrix} x_n \\ y_n \\ z_n \end{Bmatrix} \quad A(12)$$

where  $\{x_n\}^T = [x_1 \ x_2 \ \dots \ x_8]$ .

Element Stress. For linear-elastic and isotropic materials, the element stresses are calculated using Hook's law

$$\{\sigma\} = [D] \{\epsilon\} + \{\sigma^0\} \quad A(13)$$

The strain vector is  $\{\epsilon\} = [B] \{u\}$ ; therefore, the stresses are

$$\{\sigma\} = [D] [B] \{u\} + \{\sigma^0\} \quad A(14)$$

where  $\{\sigma^0\}$  is initial stress which may exist in the element. The material property matrix  $[D]$ , is defined as:

$$[D] = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ & 1-\nu & \nu & 0 & 0 & 0 \\ & & 1-\nu & 0 & 0 & 0 \\ \text{Symmetrical} & & & \frac{1-2\nu}{2} & 0 & 0 \\ & & & & \frac{1-2\nu}{2} & 0 \\ & & & & & \frac{1-2\nu}{2} \end{bmatrix} \quad A(15)$$

where E is Young's modulus and  $\nu$  is poisson's ratio for the material.

Element Equations. The potential energy  $\pi_p(u, v, w)$  which is composed of strain energy  $u_p(u, v, w)$  and  $v_p(u, v, w)$  the work done by the applied loads during displacement changes is given by [3].

$$\pi_p = \frac{1}{2} \iiint_V \{\epsilon\}^T [D] \{\epsilon\} dv - \iiint_V [F^*] \{\bar{\delta}\} dv + \iint_{s_1} [T^*] \{\bar{\delta}\} ds \quad A(16)$$

where  $[F^*] = [x^* y^* z^*]$ ,  $[T^*] = [T_x^* T_y^* T_z^*]$ .

and  $[\bar{s}] = [u, v, w]$ .

The equilibrium equations for the element are obtained by taking partial derivatives of  $\pi_p$  with respect to  $u_1, v_1, w_1$ , etc., and equating to zero,

$$\frac{\partial \pi_p}{\partial \{u\}} = 0, \quad A(17)$$

which leads to the 24 element equilibrium equations as

$$\begin{matrix} [K] & \{u\} & = & \{q\} & = & \{q_1\} & + & \{q_2\} \\ 24 \times 24 & 24 \times 1 & & 24 \times 1 & & 24 \times 1 & & 24 \times 1 \end{matrix} \quad A(18)$$

where  $[K]$  is the element stiffness matrix,

$$[k] = \iiint_V [B]^T [D] [B] dv + [K_s] \quad A(19)$$

and  $\{Q\}$  is the element nodal load vector,

$$\{Q\} = \{Q_1\} + \{Q_2\} = \iiint_V [N]^T \{F^*\} dv + \iint_{S_1} [N]^T \{T^*\} ds \quad A(20)$$

The diagonal matrix  $[K_s]$  in Eq. A(19) is the eleastic stiffness of the springs, which are connected to the boundary nodes.

### (b) Elastic-plastic analysis

Finite-element techniques applied to linear elastic materials have been solved successfully. However, for an elastic-plastic material, the coefficient in the stiffness matrix varies as a function of material loading. Two computational methods have been used successfully in the solution of elastic-plastic problems. In the first, the change at each step of load increase in plastic strain is calculated and treated as an initial strain for which the elastic stress distribution is adjusted (1). This method fails if ideal plastic is postulated or if the degree of hardening is small. In the second method, the "incremental stress method," the stress-strain relationship for every load increment is adjusted to account for plastic deformations. The work of Pope (4), Swedlow (5), Marcal and King (6), Reyes and Deere (7) and Popov and others (8) falls into this category.

The "incremental elasticity" method has one serious disadvantage. At each step of the computation the stiffness matrix of the structure is updated and iterative schemes of solution are necessary to avoid excessive computational costs. To minimize computational costs, the "initial stress" approach is used (1). In the incremental stress method, the basic elasticity matrix remains unchanged. This technique converges more rapidly than the initial strain method.

Yield Criterion. In any elastic-plastic analysis, it is necessary to introduce a yield criterion to determine the state of stress at which yielding



occurs. The von Mises yield criterion or maximum distortion energy theory of failure, which finds considerable experimental support in ductile materials, is used to determine whether the material at any point in the structure has yielded. This criterion assumes that yielding begins when the distortion energy equals the distortion energy at yield in simple tension (1). The von Mises yield criterion for a three dimensional state of stress is given by

$$F = F(\sigma) = \left[ \frac{1}{2} (\sigma_x - \sigma_y)^2 + \frac{1}{2} (\sigma_y - \sigma_z)^2 + \frac{1}{2} (\sigma_z - \sigma_x)^2 + 3\tau_{xy}^2 + 3\tau_{xz}^2 + 3\tau_{yz}^2 \right]^{1/2} - \bar{\sigma} \quad A(21)$$

where  $\bar{\sigma} = \bar{\sigma}(K)$  is the uniaxial stress at yield. If  $F(\sigma) < 0$ , the material is in elastic range. If  $F(\sigma) > 0$ , the material has experienced plastic deformation and one of the flow theories of plasticity must be used for determining the components of plastic strains and stresses due to the applied load.

During an infinitesimal increment of stress, changes of strain are assumed to be divisible into elastic and plastic parts (1). Thus, the strain increment can be written as:

$$\{d\epsilon\} = \{d\epsilon_e\} + \{d\epsilon_p\} \quad A(22)$$

where the elastic strain increments are related to the stress increments by the symmetric material matrix  $D$ . The plastic strain increments are related

to the yield criterion through Drucker's normality principle

$$\{d\epsilon_p\} = \lambda \left\{ \frac{\partial F}{\partial \sigma} \right\} \quad A(23)$$

Therefore; Eq. A(22) can be rewritten as:

$$\{d\epsilon\} = [D]^{-1} \{d\sigma\} + \lambda \left\{ \frac{\partial F}{\partial \sigma} \right\} \quad A(24)$$

At the point of incipient plasticity, the stresses are on the yield surface and the yield function is given by:

$$F(\sigma, k) = 0 \quad A(25)$$

where  $K$  is a hardening parameter.

Differentiating A(25) results in:

$$d_F = \frac{\partial F}{\partial \sigma_1} d\sigma_1 + \frac{\partial F}{\partial \sigma_2} d\sigma_2 + \dots + \frac{\partial F}{\partial k} dk = 0 \quad A(26)$$

$$\text{or } \left\{ \frac{\partial F}{\partial \sigma} \right\} T d\sigma - A \lambda = 0 \quad A(27)$$

Solving for  $A$  gives

$$A = - \frac{\partial F}{\partial k} dk \frac{1}{\lambda} \quad A(28)$$

Equations A(24) and A(27) can be written in matrix form as

$$\begin{Bmatrix} d\epsilon \\ 0 \end{Bmatrix} = \begin{bmatrix} D^{-1} & \frac{\partial F}{\partial \sigma} \\ \left(\frac{\partial F}{\partial \sigma}\right)^T & -A \end{bmatrix} \begin{Bmatrix} d\sigma \\ \lambda \end{Bmatrix} \quad A(29)$$

The constant  $\lambda$  can be eliminated from Eq. A(23). The final expression which relates the stress changes in terms of imposed strain changes can be written as:  $d\sigma = D_{ep}^* d\epsilon$

$$A(30)$$

or

$$D_{ep}^* = D - D \left\{ \frac{\partial F}{\partial \sigma} \right\} \left\{ \frac{\partial F}{\partial \sigma} \right\}^T D \left[ A + \left\{ \frac{\partial F}{\partial \sigma} \right\}^T D \left\{ \frac{\partial F}{\partial \sigma} \right\} \right]^{-1} \quad A(31)$$

where  $\left\{ \frac{\partial F}{\partial \sigma} \right\}^T = [F_x \ F_y \ F_z \ F_{xy} \ F_{yz} \ F_{xz}]$

and  $F_x = \frac{3\sigma_1}{2\bar{\sigma}}, \ F_y = \frac{3\sigma_2}{2\bar{\sigma}}, \ F_z = \frac{3\sigma_3}{2\bar{\sigma}}$

$$F_{xy} = \frac{3T_{xy}}{\bar{\sigma}}, \ F_{yz} = \frac{3T_{yz}}{\bar{\sigma}}, \ F_{zx} = \frac{3T_{zx}}{\bar{\sigma}} \quad A(32)$$

in which the dashes stand for deviatoric stresses i.e.

$$\sigma_1' = \sigma_x - \frac{(\sigma_x + \sigma_y + \sigma_z)}{3} \text{ etc.}$$

The elastic-plastic matrix  $D_{ep}^*$  replaces the elastic matrix  $D$  in incremental elastic-plastic analysis. The plastic load vector for the elements which deform plastically is given by:

$$\{dq\} = \iiint [B]^T \{d\dot{\sigma}\} dv_m \quad A(33)$$

where  $\{d\dot{\sigma}\}$  is defined as:

$$\{d\dot{\sigma}\} = \{d\sigma_e\} - \{d\sigma\} + ([De] - [Dep]) \{d\epsilon\} \quad A(34)$$

## APPENDIX B

### Description of the Finite-Element Computer Program

The computer program presented here was based on the three-dimensional 8-noded linear isoparametric cubic element. The optimum goal of this study was to develop a three-dimensional nonlinear computer program capable of extending a crack and changing the boundary conditions for the model under consideration. This program in its present form is not a general analysis program for nonlinear cracked structures. The restrictions are listed as follows: (1) the crack must lie on the x-axis and propagate in the positive x-direction, (2) the configuration and loading must be symmetric about the x-axis.

The input to the program is illustrated by using one eighth of a center-crack panel shown in Fig. 4.

#### 1. CRACK, WIDTH, THICK, HEIGHT, DAX:, SCALE (6E10.4)

The format for each input is shown in parenthesis. Crack specifies the crack length in the  $y=0$  plane. Width, thick, height represent width, thickness and height of the structure., DAX is defined as the smallest element size in the region and is used for the crack-extension in the program. Scale, scales the width, thickness and height of the specimen to the desired dimension.

#### 2. LPRIT, LMAX, KMAX, NLAYER, NEP (1615)

LPRIT = 0 indicates that no intermediate output is printed. LPRIT = 1 results in intermediate output. LMAX is the number of nodes in  $Z=0$  plane. KMAX is the number not elements in  $Z=0$  plane. NLAYER indicates the number of layers in the structure. NEP specifies elastic or

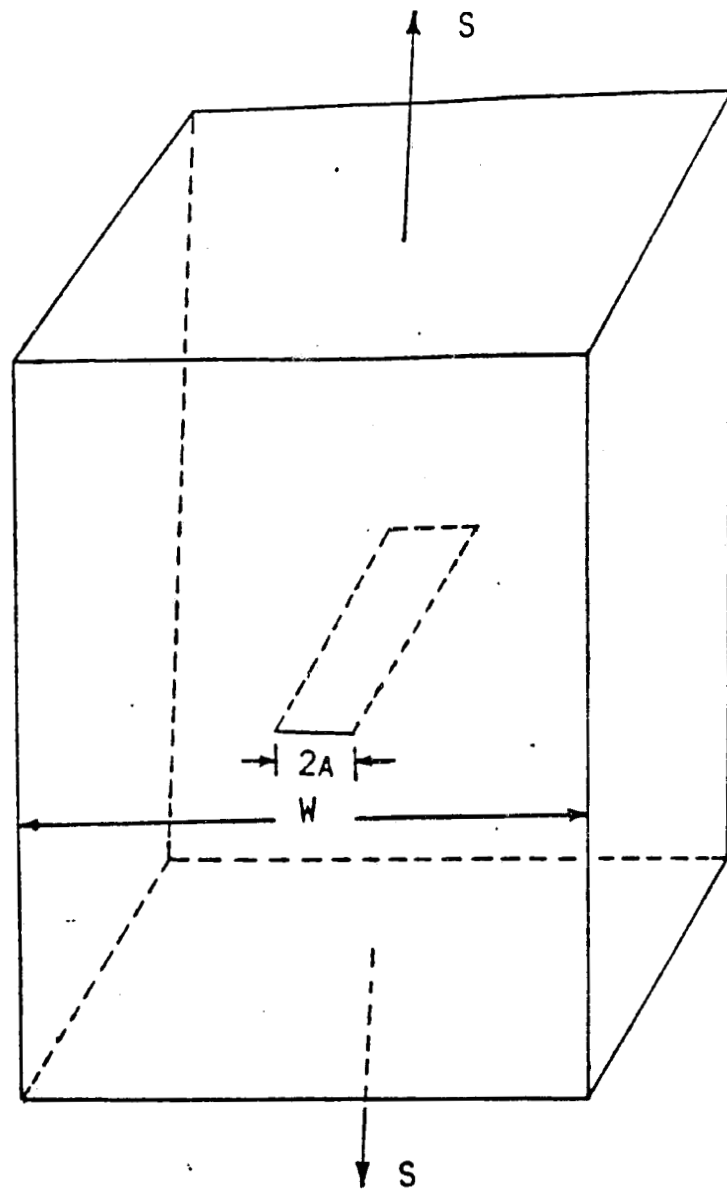


Figure 4. Center-Crack Panel subjected to uniform stress.

plastic analysis if NEP=0, elastic analysis is performed. If NEP > 0, the plastic analysis is performed.

3. K, XR(I), YR(I), ZR(I), (I5, 4X, 3 E15.7)

K refers to the node number, and XR(I), YR(I), ZR(I) are the coordinates of node K in x,y and z direction, respectively.

4. IN, (MODE(J, IN), J = 1,8) (1615)

IN describes the element number, and node gives the nodal connective of each cubic element in the structure.

5. NSYMPL (1615)

NSYMPL specifies the number of symmetric planes

6. (ISYMPY(I), I = 1, NYSMPL) (1615)

ISYMPY describes the corresponding numbers designated for each plane in the structure.

7. NFIX, NLOAD, NSPD (1615)

NFIX, NLOAD NSPD describe the number of fixed loaded, and specified displacements for nodes, respectively.

8. NODF, MU, MV, MW (1615)

NODF describes the number of fixed nodes, and MU, MV and MW represents the u, v and w displacements fixed for each node.

9. Nodlod (IL),  $P_x$ ,  $P_y$ ,  $P_z$  (1615)

Nodlod specifies the number of loaded nodes, and  $p_x$ ,  $p_y$  and  $p_z$  represent the components of loading in x, y, and z direction, respectively.

10. NODS, K, DISP(N) (1615)

NODS is the node number, K is the code for u, v and w

displacements, and Disp is the specified displacement for the corresponding node.

11. NTYP, NLM, SCRIT, RP, ACURCY (215, 4E10.4)

NTYP stands for the crack growth criterion. NLM is the number of increments to release the crack tip force. SCRIT is used for the CTOD criterion. RP is the relaxation parameter and ACURCY is used for the crack opening displacement accuracy.

12. P, WORD (E103, 1X, A<sub>4</sub>)

P designates the maximum applied stress for each cycle. The word specifies stationary or growing crack for each cycle. If word is set equal to grow, the crack will extend one element size. If word is equal to halt, the crack will be stationary for that cycle.



## APPENDIX C

### FORTRAN LISTING

ORIGINAL PAGE IS  
OF POOR QUALITY

```

PROGRAM CRACK1(INPUT,OUTPUT,TAPE7=D1,TAPE5=INPUT,
1TAPE6=OUTPUT)
COMMON/MAIN/AA(2000000),BB(9600,1),D(6,6),DINV(6,6),
1DISP(80),EPS(12400),EFEST(1550),FORCE(10),LINE(100),
2LOCAT(10),LBFOR(10),MB(9600),MSUM(9600),MPTAB(9600),
3MPLAS(12400),MODE(8,1550),MPLC(1550),NODXO(700),
4NODYO(700),NODZO(700),NODXC(700),NODYC(700),NODZC(700),
5NODFIX(80),NODLOD(80),NDISP(80),R(9600),
6SIGBAR(12400),SK(24,24),T1(9600),T2(9600),T4(2000),
7T3(9600),U(3200),UOLD(3200),V(3200),VOLD(3200),V2(3200),
8W(3200),WOLD(3200),X(74400),XR(3200),Y(74400),
9YR(3200),Z(9600),ZR(3200)
COMMON/CNST/EPST,SK2,LMAX,KMAX,DAX,
1PYLD,SCRIT,YOUNG,POIS,CRACK,PT,WIDTH,PMAX,HP,
2SBAR,LPRIT,NGAUS,NLAYER,NNODE,
3INODXO,INODYO,INODZO,INODXC,INODYC,INODZC,LNSTIF,MXNOD,
4MXNEL,MXGAUS,ICUT,LTOTB,ITNODX,KLU,NTYP,NLM,
5NDOF,KNEW,NEP,ERIT,NELM,AM,ROM
COMMON/MLTNMAT/YSTRS(20),YSTRN(20),PLMODR(20),NSEGMT
COMMON/D382/NNPE,NDF,NQD,NSTR,NQD2,NNPE2,NQD2NPE,NQD2SR,MXQ2S
COMMON/VECT/ STRV(8,6),STRSV(8,6),BMT(64,3),WDUM(8),XE(8,3),
C NCUBE(8),DIS(8,3)
DIMENSION IIMAX(3200),NSAME(3200,20),MS(8)
DIMENSION JNEW(3200),TITLE(20),ISYML(6),NBEGIN(8),NEND(8)
DIMENSION STR(6)
C *****
C * XR(I,J),YR(I,J) COORDINATES OF RECTANGULAR ELEMENTS*
C *WHICH ARE LOCATED IN THE Z=0 PLANE.
C * XR(I),YR(I),ZR(I) COORDINATES OF NODES IN THE STR*
C *UCTURE.
C *NODXO(I) NODE NUMBERS FOR PLANE X=0
C *NODYO(I) Y=0
C *NODZO(I) Z=0
C *NODXC(I) X=XCOR
C *NODYC(I) Y=YCOR
C *NODZC(I) Z=ZCOR
C *U(I),V(I),W(I) DISPLACEMENT COMPONENTS FOR EACH NODE IN THE SPECIMEN
C NODLOD(80) MAX OF 80 NODES LOADED
C *
C EPSI IS ACCURACY CHECK VALUE
C SK2 STIFFNESS OF SPRINGS CONNECTED TO BOUNDARY NODES
C LMAX NO OF NODES IN Z=0 PLANE
C KMAX NO OF ELEMENTS IN Z=0 PLANE
C DAX SMALLEST ELEMENT SIZE IN THE STRUCTURE
C PYLD LOAD AT INITIAL YIELD
C SCRIT USED FOR CTOD CRITERION
C YOUNG YOUNGS MODULUS OF THE MATERIAL
C POIS POISSON RATIO OF THE MATERIAL
C CRACK CRACK LENGTH
C PT VARIABLE USED FOR LOADING
C WIDTH WIDTH OF THE SPECIMEN
C SIGYS YIELD STRESS OF THE MATERIAL
C LPRIT LPRIT GREATER THAN 0 NO INTERNAL OUTPUT ,LPRIT=0
C INTERNAL OUTPUT(USED FOR SMALL PROBLEMS)
C NGAUS NO GAUSS POINTS IN EACH DIRECTION
C NLAYER NO OF LAYERS PUT IN THE STRUCTURE
C NNODE TOTAL NO OF NODES IN THE STRUCTURE
C INODXO TOTAL NO OF NODES IN X=0 PLANE
C INODYO TOTAL NO OF NODES IN Y=0 PLANE
C INODZO TOTAL NO OF NODES IN Z=0 PLANE
C INODXC TOTAL NO OF NODES IN X=C PLANE
C INODYC TOTAL NO OF NODES IN Y=C PLANE
C INODZC TOTAL NO OF NODES IN Z=C PLANE
C LNSTIF MAXIMUM DIMENSION FOR AA MATRIX
C MXNOD MAXIMUM NODES PUT INTO THE PROGRAM
C MXNEL MAXIMUM ELEMENTS PUT INTO THE PROGRAM

```

ORIGINAL PAGE IS  
OF POOR QUALITY

```

C      MXNOD AND MXNEL ARE FOR DIMENSIONAL PURPOSES
C      MXGAUS MAXIMUM NO OF ELEMENTS MULTIPLY THE NO OF GAUSS
C      POINTS IN EACH DIRECTION(X,Y,Z IF NGAUS=2,THE NO IS 2*2*2)
C      ICUI VARIABLE USED IN BREAK SUBPROGRAM FOR RELEASING FORCES
C      LTOTB TOTAL NO NODES IN THE THICKNESS ALONG THE CRACK TIP
C      ITNODX TOTAL NO OF NODES ALONG THE CRACK LINE
C      KLU VARIABLE USED FOR CRACK EXTENSION
C      NTYP VARIABLE USED FOR TYPE OF CRACK EXTENSION
C      NLM NO OF INCREMENTS TO RELEASE THE NODAL FORCES
C      NLOAD NO OF LOADED NODES IN THE STRUCTURE
C      NSPD NO OF SPECIFIED DISPLACEMENTS
C      MAXIT MAX NO OF ITERATION USED FOR CONVERGENCE PURPOSES
C      NDOF TOTAL NO OF DEGRES OF FREEDOM IN THE MODEL
C      NEP IF NEP =0 ELASTIC ANALYSIS,IF NEP GREATER 0 PLASTIC ANAL
C      ERIT ACCURACY CHECK VALUE FOR CONVERGENCE USED IN SUB PLAS
C      NELS TOTAL NO OF ELEMENTS IN THE SYSTEM
C      AM,ROM LINEAR OR NONLINEAR STRAIN HARDENING COEFFICIENTS
C      IF AM=0 MATERIAL IS ELASTIC-PERFECTLY .
C      KNEW VARIABLE USED IN CONTACT SUBPROGRAM TO CHECK WHETHER
C      THE NODE CLOSED OR OPENED.
C
      DATA NNPE,NDF,NQD,NSTR/8,3,2,6/
C *** OPEN MAP AND ZERO THE AA VECTOR OF LENGTH LENTOT
      LENTOT=2000000+9600*9+1550*130+700*(6)+80*4+100
      I=10*3+72+2000+3200*10+576
      J=LENTOT/65536
      JJ=LENTOT-(LENTOT/65536)*65536
      IF(JJ.NE.0) J=J+1
      LOPN=J*128
      CALL OPEN(LOPN)
C *** ZEROING THE VECTORS
      J=LENTOT/65536
      DO 223 I=1,J
      I1=(I-1)*65535+1
      AA(I1;65535)=0.0
223  CONTINUE
      J=J*65536+1
      JJ=LENTOT-J+1
      AA(J;JJ)=0.0
C
CC ***
C
      LNSTIF=2000000
      MXNEL=1550
      MXNOD=3200
      NGAUS=2
      NQD2=NGAUS**3
      NNPE2=(NNPE*(NNPE+1))/2
      NQD2NPE=NQD2*NNPE
      NQD2SR=NQD2*NSTR
      MXQ2S=MXNEL*NQD2SR
      MXGAUS=NQ2*MXNEL
      LL=3*MXNOD
      MPTAB(1;LL)=0
      Z(1;LL)=0.0
      R(1;LL)=0.0
      BB(1,1;LL)=0.0
      ZR(1;MXNOD)=0.0
      CALL Q3CLOCKS(CPU,WALL)
42  . FORMAT(I5,3F10.3)
C
C *** READ GEOMETRIC DATA
C
      READ (5,222) (TITLE(I),I=1,20)
222  FORMAT(20A4)
      WRITE(6,15) (TITLE(I),I=1,20)

```

```

15  FORMAT(1H1///5X,20A4)
    READ(5,16) CRACK,WIDTH,THICK,HEIGHT,DAX,SCALE
    WRITE(6,17) CRACK,WIDTH,THICK,HEIGHT,DAX,SCALE
16  FORMAT(6E10.4)
17  FORMAT(5X,'CRACK=',F10.4,2X,'WIDTH=',F10.4,2X,'THICK=',F10.4,
    C//5X,'HEIGHT=',F10.4,2X,'DAX=',F10.6,2X,'SCALE=',F10.5)
39  FORMAT(16I5)
    XCOR=WIDTH
    YCOR=HEIGHT
    ZCOR=THICK
    EPSI=1.E-10
    READ(5,39) LPRIT,LMAX,KMAX,NLAYER,NEP
    WRITE(6,28) LPRIT,LMAX,KMAX,NLAYER,NEP
28  FORMAT(5X,'LPR=',I2,2X,'LMAX=',I5,2X,'KMAX=',I5,2X,
    C 'NLAYER=',I2,2X,'NEP=',I2)
    NNODE=(NLAYER+1)*LMAX
    NDOF=NNODE*3
    NELM=KMAX*NLAYER
C --- CONSTANTS IN POLYNOMIAL AND D-MATRIX
    CALL ACAL
    READ (5,39) NMAT,NSEGNT
    DO 3 I=1,NMAT
    READ(5,16) YOUNG,POIS,SIGYS,AM,ROM
    WRITE(6,4) YOUNG,POIS,SIGYS,AM,ROM
    READ(5,39) (NBEGIN(IG),NEND(IG),IG=1,8)
    WRITE(6,39)(NBEGIN(IG),NEND(IG),IG=1,8)
    DO 5 IG=1,8
    IF(NBEGIN(IG).EQ.0) GOTO 3
    I1=NBEGIN(IG)*8-7
    I2=NEND(IG)*8-I1+1
5    SIGBAR(I1,I2)=SIGYS
3    CONTINUE
4    FORMAT(//10X,'MODULUS, NUE, YIELD STRESS, AM, & ROM:',5E12.4)
    CALL DCON(YOUNG,POIS,D,DINV)
C
C *** READ COORDINATES AND CONNECTIVITY
C
    DO 30 I=1,NNODE
    JNEW(I)=I
30  READ(7,20) K,XR(I),YR(I),ZR(I)
20  FORMAT(15,4X,3E15.7)
    WRITE(6,333)
    WRITE(6,861)(J,XR(J),YR(J),ZR(J),J=1,NNODE)
861  FORMAT(2(3X,I5,3(E13.6,1X)))
333  FORMAT(1H1//10X,'NODAL COORDINATES,NODE#, X,Y,AND,Z'//)
    DO 31 IE=1,NELM
31  READ(7,39) IN,(MODE(J,IN),J=1,8)
    WRITE(6,334)
334  FORMAT(1H1//5X,'NODAL CONNECTIVITY IE, I,J,K,L, I1,J1,K1,L1'//)
    WRITE(6,864) (IE,(MODE(J,IE),J=1,8),IE=1,NELM)
864  FORMAT(2(5X,9I5))
C
C ***
C
    IZIPI=5
    CALL Q3CLOCKS(CPU,WALL)
    WRITE(6,9999) IZIPI,CPU,WALL
9999  FORMAT(5X,'STEP#',I3,2X,'TIME IN SECS: CPU=',F10.4,2X,
    C 'WALL=',F12.3)
    WRITE(6,1607) NELM
1607  FORMAT(5X,'TOTAL NO OF HEXAHEDRAN=',I6)
C *** IDENTIFY NODES ON CONSTANTS PLANES
C    IDENTIFY X=0 PLANE ,STORE NODX0 ARRAY
C    IDENTIFY Y=0 PLANE ,STORE NODY0 ARRAY
C    IDENTIFY Z=0 PLANE ,STORE NODZ0 ARRAY
C    IDENTIFY X=C PLANE ,STORE NODXC ARRAY

```

```

C          IDENTIFY Y=C PLANE ,STORE NODYC ARRAY
C          IDENTIFY Z=C PLANE ,STORE NODZC ARRAY
C
      INODXO=0
      INODYO=0
      INODZO=0
      INODXC=0
      INODYC=0
      INODZC=0
      DO 1300 I=1,NNODE
      IF(ABS(XR(I)).LE.EPSI) GO TO 1301
      DX1=ABS(XR(I)-XCOR)
      IF(DX1.GT.EPSI) GO TO 1302
      INODXC=INODXC+1
      NODXC(INODXC)=I
      GO TO 1302
1301      INODXO=INODXC+1
      NODXO(INODXO)=I
1302      IF(ABS(YR(I)).LE.EPSI) GO TO 1303
      DY1=ABS(YR(I)-YCOR)
      IF(DY1.GT.EPSI) GO TO 1304
      INODYC=INODYC+1
      NODYC(INODYC)=I
      GO TO 1304
1303      INODYO=INODYO+1
      NODYO(INODYO)=I
1304      IF(ABS(ZR(I)).LE.EPSI) GO TO 1305
      DZ1=ABS(ZR(I)-ZCOR)
      IF(DZ1.GT.EPSI) GO TO 1300
      INODZC=INODZC+1
      NODZC(INODZC)=I
      GO TO 1300
1305      INODZO=INODZO+1
      NODZO(INODZO)=I
1300      CONTINUE
      WRITE(6,1002)
1002      FORMAT(5X,' INODXO, 5X, INODYO, 5X, INODZO, 5X, INODXC, 5X, INODYC
1, 5X, INODZC')
      WRITE(6,1122) INODXO, INODYO, INODZO, INODXC, INODYC, INODZC
1122      FORMAT(8X,6I6)
      IF(LPRIT.EQ.0) WRITE(6,39) (NODXO(I),I=1,INODXO)
      IF(LPRIT.EQ.0) WRITE(6,39) (NODYO(I),I=1,INODYO)
      IF(LPRIT.EQ.0) WRITE(6,39) (NODZO(I),I=1,INODZO)
      IF(LPRIT.EQ.0) WRITE(6,39) (NODXC(I),I=1,INODXC)
      IF(LPRIT.EQ.0) WRITE(6,39) (NODYC(I),I=1,INODYC)
      IF(LPRIT.EQ.0) WRITE(6,39) (NODZC(I),I=1,INODZC)
1001      CONTINUE
      IZIPL=10
      CALL Q3CLOCKS(CPU,WALL)
      WRITE(6,9999) IZIPL,CPU,WALL
C
C *****
C
      CALL NSAMC(MODE,NSAME,IIMAX,MB,NNODE,NELM,8,MXNOD,MXNEL,NDOF)
      MSUM(1)=0
      MSUM(2)=1
      DO 352 I=3,NDOF
      LN=I-1
352      MSUM(I)=MSUM(LN)+MB(LN)
      LDOF=MSUM(NDOF)+MB(NDOF)
      WRITE(6,504) LDOF
504      FORMAT('//10X,'STORAGE REQUIREMENT FOR STIFFNESS MATRIX IS='110)
      IZIPL=14
      CALL Q3CLOCKS(CPU,WALL)
      WRITE(6,9999) IZIPL,CPU,WALL
C

```

```

      SK2=YOUNG*1.0E+07
C
C   ASSEMBLE THE STIFFNESS MATRIX K
C
      DO 943 I=1,NELM
      NCUBE(1;8)=MODE(1,I;8)
      MS(1;8)=NCUBE(1;8)
      CALL CORDIN(NCUBE,MXNOD,XR,YR,ZR,XE)
      CALL SMALLK(SK,XE,D,IERR)
      DO 943 J=1,8
      DO 943 L=1,8
        IF(MS(L).LT.MS(J)) GO TO 943
        IU=3*MS(J)-2
        IV=IU+1
        IW=IV+1
        JU=3*MS(L)-2
        JV=JU+1
        JW=JV+1
        N1=MSUM(JU)-JU+MB(JU)+IU
        N2=N1+1
        N3=N2+1
        N4=MSUM(JV)-JV+MB(JV)+IU
        N5=N4+1
        N6=N5+1
        N7=MSUM(JW)-JW+MB(JW)+IU
        N8=N7+1
        N9=N8+1
        MC1=3*J-2
        MC2=MC1+1
        MC3=MC2+1
        MR1=3*L-2
        MR2=MR1+1
        MR3=MR2+1
        AA(N1)=AA(N1)+SK(MR1,MC1)
        AA(N4)=AA(N4)+SK(MR2,MC1)
        AA(N5)=AA(N5)+SK(MR2,MC2)
        AA(N7)=AA(N7)+SK(MR3,MC1)
        AA(N8)=AA(N8)+SK(MR3,MC2)
        AA(N9)=AA(N9)+SK(MR3,MC3)
952      IF(J.EQ.L) GO TO 943
        AA(N2)=AA(N2)+SK(MR1,MC2)
        AA(N3)=AA(N3)+SK(MR1,MC3)
        AA(N6)=AA(N6)+SK(MR2,MC3)
943      CONTINUE
        IZIP1=18
        CALL Q3CLOCKS(CPU WALL)
        WRITE(6,9999) IZIP1,CPU,WALL
C
C *** IMPOSE SYMMETRIC BOUNDARY CONDITIONS
C
      READ (5,39) NSYMP
      WRITE(6,315) NSYMP
315      FORMAT(/5X,' # OF SYMMETRIC BOUNDARY CONDITIONS =',I3)
      IF(NSYMP.EQ.0) GOTO 314
      READ(5,39) (ISYMP(I),I=1,NSYMP)
      WRITE(6,316) (ISYMP(I),I=1,NSYMP)
316      FORMAT(10X,' SYMMETRIC PLANE NUMBERS ARE :',6I3)
      DO 317 IS=1,NSYMP
        ISY=ISYMP(IS)
        IF(ISY.EQ.1) CALL SYMPLN(AA,MSUM,MB,MPTAB,NODXO,INODXO,SK2,1,
          CNDOF,LNSTIF)
        IF(ISY.EQ.2) CALL SYMPLN(AA,MSUM,MB,MPTAB,NODYO,INODYO,SK2,2,
          CNDOF,LNSTIF)
        IF(ISY.EQ.3) CALL SYMPLN(AA,MSUM,MB,MPTAB,NODZO,INODZO,SK2,3,
          CNDOF,LNSTIF)
        IF(ISY.EQ.4) CALL SYMPLN(AA,MSUM,MB,MPTAB,NODXC,INODXC,SK2,1,

```

```

CND0F, LNSTIF)
  IF (ISY.EQ.5) CALL SYMPLN(AA,MSUM,MB,MPTAB,NODYC,INODYC,SK2,2,
CND0F, LNSTIF)
  IF (ISY.EQ.6) CALL SYMPLN(AA,MSUM,MB,MPTAB,NODZC,INODZC,SK2,3,
CND0F, LNSTIF)
317 . CONTINUE
314 . CONTINUE
C
C *** SYMMETRIC BOUNDARY CONDITIONS ON THE CRACK PLANE
C
      DO 318 I=1, INODYO
      L=NODYO(I)
      SAP=XR(L)
      IF (SAP.LT.CRACK) GO TO 318
      NV=3*L-1
      NVNV=MSUM(NV)+MB(NV)
      MPTAB(NV)=MV
      AA(NVNV)=AA(NVNV)+SK2
318 . CONTINUE
C
      IZIPI=27
      CALL Q3CLOCKS(CPU,WALL)
      WRITE(6,9999) IZIPI,CPU,WALL
C ***** READ BOUNDARY CONDITIONS AND LOADING
C
C *** FIXED NODES AND LOADING
C
      READ(5,39) NFIX,NLOAD,NSPD
      WRITE(6,40) NFIX,NLOAD,NSPD
40 . FORMAT(//5X,'# OF NODES: FIXED=',I3,2X,'LOADED=',I3,2X,
C 'SP. DISP=',I3//)
      IF(NFIX.EQ.0) GOTO 417
      DO 416 IFIX=1,NFIX
      READ(5,39) NODF,MU,MV,MW
      WRITE(6,39) NODF,MU,MV,MW
      NODFIX(IFIX)=JNEW(NODF)
      NU=JNEW(NODF)*3-2
      NUNU=MSUM(NU)+MB(NU)
      NVNV=MSUM(NU+1)+MB(NU+1)
      NWNW=MSUM(NU+2)+MB(NU+2)
      AA(NUNU)=AA(NUNU)+MU*SK2
      AA(NVNV)=AA(NVNV)+MV*SK2
      AA(NWNW)=AA(NWNW)+MW*SK2
      MPTAB(NU)=MU
      MPTAB(NU+1)=MV
416 . MPTAB(NU+2)=MW
417 . CONTINUE
      IF(NLOAD.LE.0) GOTO 739
      DO 41 IL=1,NLOAD
      READ(5,42) NODLOD(IL),PX,PY,PZ
      IZ=NODLOD(IL)
      WRITE(6,43) NODLOD(IL),PX,PY,PZ
      NODLOD(IL)=JNEW(IZ)
      IZ1=(JNEW(IZ)-1)*3+1
      BB(IZ1,1)=PX
      BB(IZ1+1,1)=PY
      BB(IZ1+2,1)=PZ
41 . CONTINUE
43 . FORMAT(5X,I5,3(F12.5,2X))
739 . IF(NSPD.LE.0) GOTO 738
      DO 735 N=1,NSPD
      READ(5,736) NODS,K,DISP(N)
      WRITE(6,737) NODS,K,DISP(N)
      NU=(JNEW(NODS)-1)*3+K
      NDISP(N)=NU
      NUNU=MSUM(NU)+MB(NU)

```

```

      AA(NUNU)=SK2
735  BB(NU,1)=SK2*DISP(N)
736  FORMAT(2I5,E14.5)
737  FORMAT(5X,I5,2X,I1,3X,E12.5)
738  CONTINUE
      R(1;NDOF)=BB(1,1;NDOF)
      IZIP1=16
      CALL Q3CLOCKS(CPU,WALL)
      WRITE(6,9999) IZIP1,CPU,WALL
      IFAC=0
      ALP=0
      IZIP1=21
      CALL SYMBAN(LNSTIF,NDOF,MB,MSUM,AA,1,BB,IFAC,T1,IERR
1,ALP,Z,T2,T3,T4,1)
      IZIP1=22
      CALL Q3CLOCKS(CPU,WALL)
      WRITE(6,9999) IZIP1,CPU,WALL
      IF(IERR.EQ.1) WRITE(6,415) IERR
415  FORMAT(/10X'IERR='I2,10X'NONPOSITIVE DEFINITE MATRIX')
      IF(IERR.NE.0) STOP
C      PRINT OUT UNIT LOAD DISPLACEMENTS AND STRESSES.
C
9000  CONTINUE
      WRITE(6,425)
425  FORMAT(1H1///10X,'UNIT LOAD DISPLACEMENTS AND STRESSES'//)
      WRITE(6,418)
418  FORMAT(6X,4HNODE,6X,1HX,15X,1HY,13X,1HZ,13X,1HU,13X,
1 1HV,13X,1HW/)
      DO 551 N=1,NNODE
551  IIMAX(N)=(N-1)*3+1
      U(1;NNODE)= Q8VGATHR(BB(1,1;NDOF),IIMAX(1;NNODE);U(1;NNODE))
      IIMAX(1;NNODE)=IIMAX(1;NNODE)+1
      V(1;NNODE)= Q8VGATHR(BB(1,1;NDOF),IIMAX(1;NNODE);V(1;NNODE))
      IIMAX(1;NNODE)=IIMAX(1;NNODE)+1
      W(1;NNODE)= Q8VGATHR(BB(1,1;NDOF),IIMAX(1;NNODE);W(1;NNODE))
      DO 944 IN=1,NNODE
944  WRITE(6,420) IN,XR(IN),YR(IN),ZR(IN),U(IN),V(IN),W(IN)
420  FORMAT(5X,I5,2X,3(2X,E11.5),3(2X,E12.4))
C
C ***
C
      PYLD=0.0
      WRITE(6,306)
306  FORMAT(1H1///10X,'ELASTIC STRESSES: SX, SY, SZ, AND SYZ, SZX, SXY')
307  FORMAT(5X,I6,2X,6E12.4)
      IGAUSP=0
      DO 300 IE=1,NELM
      NCUBE(1;8)=MODE(1,IE;8)
      DO 301 I=1,8
      I1=NCUBE(I)
      DIS(I,1)=U(I1)
      DIS(I,2)=V(I1)
301  DIS(I,3)=W(I1)
      CALL CORDIN(NCUBE,MXNOD,XR,YR,ZR,XE)
      CALL STRESS(DIS,XE,D,STRV,STRSV,BMT,WDUM)

      ILOC=(IE-1)*NQD2SR+1
      X(ILOC;NQD2SR)=STRSV(1,1;NQD2SR)
      Y(ILOC;NQD2SR)=STRV(1,1;NQD2SR)
      DO 350 IG=1,NQD2
      DO 360 J=1,6
360  STR(J)=STRSV(IG,J)
      IGAUSP=IGAUSP+1
      CALL SEQU(STR,SEFF)
      SEFF=SEFF/SIGBAR(IGAUSP)
      IF(PYLD.GT.SEFF) GOTO 350

```



```

        PYLD=SEFF
        IEY=IE
        IGAUSY=IG
350    CONTINUE
        WRITE(6,307) IE,(STR(J),J=1,6)
300    CONTINUE
        PYLD=1./PYLD
        WRITE(6,305) IEY,IGAUSY,PYLD
305    FORMAT(1H1//10X,"ELEMENT#",I5,2X,"GAUSS PT=",I2,2X,"LOAD FACTOR AT
C ", "YIELD=", E12.6)
        READ(5,450) NTYP,NLM,SCRIT,RP,ACURCY
        WRITE(6,412) NTYP,SCRIT,NLM,RP,ACURCY
412    FORMAT(/9X,"CRACK GROWTH CRITERION NTYP=",I2," AND CTOD =",E10.4,
C//10X,"NUMBER OF INCREMENTS TO RELEASE CRACK TIP FORCE=",I2,
C//10X,"RELAXATION PARAMETER=",F5.2,"(NORMAL)",
C//10X,"CRACK OPENING DISPLACEMENT ACCURCY=",E12.4)
450    FORMAT(2I5,4E10.4)
        IF(NEP.EQ.0) STOP
        CALL PLAS
        IZIP1=26
9991    STOP
        END
        SUBROUTINE OPEN(LOPN)
        COMMON/MAIN/AA(2000000),BB(9600,1),D(6,6),DINV(6,6),
1DISP(80),EPS(12400),EFEST(1550),FORCE(10),LINE(100),
2LOCAT(10),LBFOR(10),MB(9600),MSUM(9600),MPTAB(9600),
3MPLAS(12400),MODE(8,1550),MPLC(1550),NODXO(700),
4NODYO(700),NODZO(700),NODXC(700),NODYC(700),NODZC(700),
5NODFIX(80),NODLOD(80),NDISP(80),R(9600),
6SIGBAR(12400),SK(24,24),T1(9600),T2(9600),T4(2000),
7T3(9600),U(3200),UOLD(3200),V(3200),VOLD(3200),V2(3200),
8W(3200),WOLD(3200),X(74400),XR(3200),Y(74400),
9YR(3200),Z(9600),ZR(3200)
        COMMON/CNST/EPSI,SK2,LMAX,KMAX,DAX,
1PYLD,SCRIT,YOUNG,POIS,CRACK,PT,WIDTH,FMAX,HP,
2SBAR,LPRIT,NGAUS,NLAYER,NNODE,
3INODXO,INODYO,INODZO,INODXC,INODYC,INODZC,LNSTIF,MXNOD,
4MXNEL,MXGAUS,ICUT,LTOTB,ITNODX,KLU,NTYP,NLM,
5NDOF,KNEW,NEP,ERIT,NELM,AM,ROM
C
C        THIS SUB-PROGRAM OPENS ALL THE Q3OPNMAP FILES
C        MAXIMUM LENGTH OF ANY FILE IS 5376 SMALL PAGES (DECIMAL)
C
C        TO CHANGE THE MAXIMUM LENGTH CHANGE THE DATA CARD
C        DATA LMAX /      /
C
        CHARACTER*8 FILE, WORD(8)
        DATA WORD/ "ASTIF001", "ASTIF002",
Z          "ASTIF003", "ASTIF004",
Z          "ASTIF005", "ASTIF006",
Z          "ASTIF007", "ASTIF008" /
        DATA LMAX / 5376/
        IF(LOPN.LE. LMAX) GO TO 20
        LOPNA= LMAX
        LDUM= LOPN/LMAX
        LOPNB= LOPN-LDUM*LMAX
        DO 10 I=1,LDUM
        FILE= WORD(I)
        ISTART= LMAX*512*(I-1)+1
        CALL Q3OPNMAP (IERR, FILE, AA(ISTART), LOPNA, 1)
        PRINT 100, IERR, FILE, LOPNA
        WRITE(6, 100) IERR, FILE, LOPNA
        IF(IERR.NE.0) STOP
100    FORMAT (10X," IERR FROM OPNMAP=",Z16,5X, " FILE ",A8,
Z          2X," IS OF LENGTH ",I10,2X," SMALL PAGES (DECIMAL)",/)
10    CONTINUE

```

```

      IF( LOPNB.EQ.0) RETURN
      ISTART= LMAX*LDUM*512+1
      FILE= WORD( LDUM+1)
      CALL Q3OPNMAP (IERR, FILE, AA(ISTART), LOPNB, 1)
      PRINT 100, IERR, FILE, LOPNB
      WRITE(6, 100) IERR, FILE, LOPNB
      IF(IERR.NE.0) STOP
      RETURN
20    CONTINUE
      FILE= WORD(1)
      CALL Q3OPNMAP ( IERR, FILE, AA(1), LOPN, 1)
      PRINT 100, IERR, FILE, LOPN
      WRITE(6, 100) IERR, FILE, LOPN
      IF(IERR.NE.0) STOP
      RETURN
      END
      FUNCTION FNMAT(SBAR,CE,EPS,M)
      COMMON/MLTNMAT/YSTRS(20),YSTRN(20),PLMODR(20),NSEGMT
C
C --- EPST= TOTAL STRAIN
C --- EPS = PLASTIC STRAIN
C
      EPST=EPS+SBAR/CE
      DO 10 I=1,NSEGMT
10    IF(EPST.LT.YSTRN(I)) GOTO 11
11    FNMAT=PLMODR(I)*CE
      RETURN
      END
      SUBROUTINE EEMPLN(AA,MSUM,MB,MPTAB,NODP,INOD,SK2,ID,NDOF,LNSTIF)
C
C *** IMPOSING SYMMETRIC BOUNDARY CONDITIONS
C
      DIMENSION AA(LNSTIF),MSUM(NDOF),MB(NDOF),MPTAB(NDOF),NODP(INOD)
      DO 100 I=1,INOD
      L=NODP(I)
      NU=(L-1)*3+ID
      NUNU=MSUM(NU)+MB(NU)
      MPTAB(NU)=NU
100   AA(NUNU)=AA(NUNU)+SK2
      RETURN
      END
      SUBROUTINE NSAMC(MSAME,NSAME,IIMAX,MB,LMAX,KMAX,NODPEL,MXNOD,
      IMXNEL,NDOF)
      DIMENSION MSAME(NODPEL,MXNEL),NSAME(MXNOD,20),IIMAX(MXNOD),
      C MB(NDOF)
C *****
C   MXNEL = MAXIMUM NUMBER OF ELEMENTS
C   MXNOD = MAXIMUM NUMBER OF NODES
C   NODPEL = # OF NOED PER ELEMENTS
C   LMAX = # OF NOEDS IN THE PROBLEM
C   KMAX = # OF ELEMENTS IN THE PROBLEM
C   MSAME(NODPEL,IEL) = NOEDL CONNECTIVITY IF IEL ELEMENT
C   NDOF = LMAX* # OF DOF PER NODE
C   MB(NDOF) = BAND WIDTHS OF ALL NDOF DEGREE-OF- FREEDOM
C *****
      DO 10 IE=1,KMAX
      DO 20 J=1,NODPEL
      IK=MSAME(J,IE)
      IIMAX(IK)=IIMAX(IK)+1
20    NSAME(IK,IIMAX(IK))=IE
10    CONTINUE
C16  FORMAT(16I5)
C   ***CALCULATE MB VECTOR
      IBANDW=0
      DO 350 NODE=1,LMAX

```

```

      MAXDIF=0
      IM=IIMAX(NODE)
      DO 351 M=1,IM
      NTRI=NSAME(NODE,M)
      DO 351 L=1,NODPEL
      NUM=MSAME(L,NTRI)
      NDIFF=3*(NUM-NODE)
      IF(NDIFF.LT.MAXDIF) MAXDIF=NDIFF
351  CONTINUE
      IF(IBANDW.LT.IABS(MAXDIF)) IBANDW=IABS(MAXDIF)
      NU=3*(NODE-1)+1
      NV=NU+1
      NW=NV+1
      MB(NU)=IABS(MAXDIF)+1
      IF(MB(NU).GT.NU) MB(NU)=NU
      MB(NV)=MB(NU)+1
      MB(NW)=MB(NV)+1
350  CONTINUE
      IBANDW=IBANDW+3
      WRITE(6,25) IBANDW
25.  FORMAT(5X,'MAX BAND WIDTH=',I6)
      RETURN
      END
      SUBROUTINE DCON(YOUNG,POIS,D,DINV)
C
C *** 3-D D(6,6) & DINV MATRICES FOR ISOTROPIC MATERIAL
C
      DIMENSION D(6,6),DINV(6,6)
      DEL=YOUNG*(1-POIS)/((1+POIS)*(1-2*POIS))
      DEL2=POIS/(1-POIS)
      DEL3=(1-2*POIS)/(2*(1-POIS))
      D(1,1;36)=0.0
      DINV(1,1;36)=0.0
      D(1,1)=DEL
      D(1,2)=DEL*DEL2
      D(1,3)=D(1,2)
      D(2,2)=D(1,1)
      D(2,3)=D(1,3)
      D(3,3)=D(1,1)
      D(4,4)=DEL*DEL3
      D(5,5)=D(4,4)
      D(6,6)=D(5,5)
C *** INVERSE OF D-MATRIX
      DINV(1,1)=1./YOUNG
      DINV(1,2)=-POIS/YOUNG
      DINV(1,3)=DINV(1,2)
      DINV(2,2)=DINV(1,1)
      DINV(2,3)=DINV(1,2)
      DINV(3,3)=DINV(1,1)
      DINV(4,4)=2.*(1+POIS)/YOUNG
      DINV(5,5)=DINV(4,4)
      DINV(6,6)=DINV(4,4)
      DO 5 I=1,3
      DO 5 J=I,3
      D(J,I)=D(I,J)
      DINV(J,I)=DINV(I,J)
5      CONTINUE
      RETURN
      END
      SUBROUTINE SHAPE(X,Y,Z,R)
C
C *** SHAPE FUNCTIONS
C
      DIMENSION R(8)
      R(1)=1.
      R(2)=X

```

```

      R(3)=Y
      R(4)=Z
      R(5)=X*Y
      R(6)=Y*Z
      R(7)=Z*X
      R(8)=X*Y*Z
      RETURN
      END
      SUBROUTINE CORDIN(NCUBE,MXNOD,XR,YR,ZR,A)
C
C *** EVALUATE A(8,3) CARTESIAN COORDINATE MATRIX
C
      DIMENSION A(8,3),NCUBE(8),XR(MXNOD),YR(MXNOD),ZR(MXNOD)
      DO 1 I=1,8
      N1=NCUBE(I)
      A(I,1)=XR(N1)
      A(I,2)=YR(N1)
      A(I,3)=ZR(N1)
1      RETURN
      END
      SUBROUTINE ACAL
      COMMON/AINV/AI(8,8)
      COMMON/GENRL/GCR(8,3)
      DIMENSION R1(8),DUM(8,1),IPIVOT(8),IWK(16),A2(8,8)
      A2(1,1;64)=0.0
      DO 1 I=1,8
      X1=GCR(I,1)
      Y1=GCR(I,2)
      Z1=GCR(I,3)
      CALL SHAPE(X1,Y1,Z1,R1)
      DO 1 J=1,8
1      A2(I,J)=R1(J)
      CALL MATINV(A2,8,8,DUM,1,0,DET)
      AI(1,1;64)=A2(1,1;64)
      RETURN
      END
      SUBROUTINE DERIVE (X,Y,Z,R)
      COMMON/AINV/AI(8,8)
      ROWWISE DN(3,8)
      DIMENSION R(3,8)
      DN(1,1;24)=0.0
C****      DN/DXI NOW
      DN(1,2)=1.0
      DN(1,5)=Y
      DN(1,7)=Z
      DN(1,8)=Y*Z
C****      DN/DETA NOW
      DN(2,3)=1.0
      DN(2,5)=X
      DN(2,6)=Z
      DN(2,8)= X*Z
C****      DN/DZETA NOW
      DN(3,4)=1.0
      DN(3,6)=Y
      DN(3,7)=X
      DN(3,8)=X*Y
      DO 10 J=1,3
      DO 10 I=1,8
      R(J,I)= Q8SDOT ( DN(J,1;8) , AI(1,I;8) )
10      CONTINUE
      RETURN
      END
      SUBROUTINE SMALLK( SMK, XE, D, IERR)
C
C THIS MODULE GENERATES AN ELEMENTAL STIFFNESS MATRIX FOR THE GIVEN
C ELEMENT. VECTOR VERSION

```

```

C
C   DIMENSION SMK(24,24), XE(8,3), D(6,6)
C   COMMON/D382/NNPE,NDF,NQD,NSTR,NQD2,NNPE2,NQD2NPE,NQD2SR,MXQ2S
C   REAL KE
C   DIMENSION KE(324)
C
C
C INPUT:   D(6,6) = MODULUS MATRIX
C          XE(8,3) = 8 NODES X,Y, Z COORDINATES
C OUTPUT :  SMK(24,24) = STIFFNESS MATRIX
C
C
C KE       - ELEMENTAL STIFFNESS MATRICES FOR ALL DISTINCT ELEMENTS, IN
C           ROWWISE NODAL BLOCK LOWER TRIANGULAR FORM
C PE       - ELEMENTAL LOAD VECTORS FOR ALL ELEMENTS IN NODAL BLOCK FORM
C DATA NDFX, NSTRX, NNPEX / 3, 6, 8 /
C
C
C NDF      - NUMBER OF DISPL. DEGREES OF FREEDOM PER NODE
C NSTR     - NUMBER OF STRESS RESULTANTS PER NODE
C NQD      - NUMBER OF QUADRATURE POINTS IN EACH DIRECTION
C NNPE     - NUMBER OF NODES PER ELEMENT
C
C
C***      ETH= THERMAL STRAINS IN THE CARTESIAN SYSTEM.
C****     FTHERM= THERMAL LOAD VECTOR.
C
C
C [D] - STRESS STRAIN MATRIX
C
C   DIMENSION IBSP(6,3), B(64,3), BJ(288,3), CK(288,3),
C   Z   WTDTEX(64), SUM(288)
C   DIMENSION WTDTEX(8)
C   DIMENSION CTH(64), TPST(8)
C   DATA IBSP/ 1, 2*0, 3, 0, 3,
C   Z   0, 2, 0, 1, 3, 0,
C   Z   2*0, 3, 0, 2, 1 /
C
C [IBSP]   - SPARSITY PATTERN AND POINTER MATRIX FOR [B] AND [BJ]
C [B]      - STRAIN DISPLACEMENT MATRIX
C [BJ]     - ANOTHER STRAIN DISPLACEMENT MATRIX
C [CK]     - A ROW FOR EACH STIFFNESS MATRIX NODAL PARTITION
C (WTDTEX) - REPLICATED WEIGHTED DETERMINANTS
C (SUM)    - TEMPORARY STORAGE
C
C   DIMENSION IREPL(36), IPOSN(210)
C   DESCRIPTOR IREPLD, SORCD, DESTD
C
C IREPL    - VECTOR OF LENGTH "NNPE2" CONTAINING ZEROS USED IN THE
C           REPLICATION PROCESS
C IPOSN    - ARRAY OF LENGTH "NNPE2" USED TO CORRECTLY POSITION
C           THE NODAL PARTITIONS IN [KE]
C IREPLD   - VECTOR DESCRIPTOR FOR (IREPL)
C SORCD    - VECTOR DESCRIPTOR FOR THE REPLICATION SOURCE
C DESTD    - VECTOR DESCRIPTOR FOR THE REPLICATION DESTINATION
C
C   DATA LENI, LENB, LENBJ, LENC, LENW, LENWT
C   Z   / 18, 192, 864, 864, 288, 64 /
C
C THESE ARE THE DIMENSIONED LENGTHS OF [IBSP], [B], [BJ], [CK],
C (WTDTEX), AND (SUM) FOR ZEROING OUT PURPOSES.
C

```

```

      L3 = NQD2NPE - NQD2 + 1
      DO 140 KK = 1, NNPE
        ASSIGN IREPLD, IREPL(1; KK)
        ASSIGN SORCD, B(L3,IT; NQD2)
        ASSIGN DESTD, BJ(L1,IT; L2)
        CALL Q8VXTOV(X'02', 0, IREPLD, 0, SORCD, 0, DESTD)
        L2 = L2 + NQD2
        L1 = L1 - L2
        L3 = L3 - NQD2
140      CONTINUE
150      CONTINUE
C
      DO 155 IT=1,3
        B(1,IT;NQD2NPE) = B(1,IT;NQD2NPE)*WDETTEX(1;NQD2NPE)
155      CONTINUE
C
C FORM ([B]**T * [D]) * [BJ] A ROW AT A TIME.
C
      L9=1
      DO 400 II = 1, NDF
        CK(1,1; LENDF) = 0.
        DO 230 KK = 1, NSTR
          SUM(1; NQD2NPE) = 0.
          DO 210 JJ = 1, NSTR
            IT = IBSP(JJ,II)
            IF (IT .EQ. 0) GOTO 210
            IF (D(JJ,KK) .EQ. 0.) GOTO 210
            SUM(1; NQD2NPE) = SUM(1; NQD2NPE) + B(1,IT; NQD2NPE)
                                                    * D(JJ,KK)
210          CONTINUE
C
C FILL UP THE REST OF (SUM)
C
      IF (SUM(1) .EQ. 0.) GOTO 225
      L1 = LE - NQD2 + 1
      L2 = NQD2
      L3 = NQD2NPE - NQD2 + 1
      DO 215 JJ = 2, NNPE
        SUM(L1; L2) = SUM(L3; L2)
        L2 = L2 + NQD2
        L1 = L1 - L2
        L3 = L3 - NQD2
215      CONTINUE
      DO 220 JJ = 1, NDF
        IT = IBSP(KK,JJ)
        IF (IT .EQ. 0) GOTO 220
        CK(1,JJ; LE) = CK(1,JJ; LE) + SUM(1; LE) * BJ(1,IT; LE)
220      CONTINUE
225      CONTINUE
230      CONTINUE
C
C WE NOW HAVE THE II-TH ROW (BEFORE SUMMING) FOR ALL
C "NNPE2" NODAL PARTITIONS OF THE ELEMENTAL STIFFNESS MATRIX.
C
      DO 310 JJ = 1, NDF
        L1 = 1
        DO 300 KK = 1, NNPE2
          L2 = L9 + IPOSN(KK)
          KE(L2) = Q8SSUM(CK(L1,JJ; NQD2))
          L1 = L1 + NQD2
300      CONTINUE
        L9 = L9 + 1
310      CONTINUE
400      CONTINUE
C
      L9=1

```

```

Z , (B(6401),DAJ23(1))
Z , (B(7681),DAJ31(1))
Z , (B(8961),DAJ32(1))
Z , (B(10241),DAJ33(1))
Z , (B(11521),AJ11 (1 ))
Z , (B(11585),AJ12 (1 ))
Z , (B(11649),AJ13 (1 ))
Z , (B(11713),AJ21 (1 ))
Z , (B(11777),AJ22 (1 ))
Z , (B(11841),AJ23 (1 ))
Z , (B(11905),AJ31 (1 ))
Z , (B(11969),AJ32 (1 ))
Z , (B(12033),AJ33 (1 ))
Z , (B(12097),AJI11(1))
EQUIVALENCE (B(12161),AJI12(1))
Z , (B(12225),AJI13(1))
Z , (B(12289),AJI21(1))
Z , (B(12353),AJI22(1))
Z , (B(12417),AJI23(1))
Z , (B(12481),AJI31(1))
Z , (B(12545),AJI32(1))
Z , (B(12609),AJI33(1))
Z , (B(12673),DET11(1))
Z , (B(12737),DET12(1))
Z , (B(12801),DET13(1))
Z , (B(12865),DET21(1))
Z , (B(12929),DET22(1))
Z , (B(12993),DET23(1))
Z , (B(13057),DET31(1))
Z , (B(13121),DET32(1))
Z , (B(13185),DET33(1))
C DIMENSION BJ(216,3)
C**** THE ABOVE DIMENSIONS ALLOW UPTO 4 POINT GAUSSIAN IN EACH DIRECTION
LEN=11520
LENI=13248
CALL ZEROLV(BB,LEN)
CALL ZEROLV(B,LENI)
NG= N*N*N
C*** NG ARE THE TOTAL NO OF INTEGRATION POINTS
NS=8
C**** NS= NO OF SHAPE FUNCTIONS
NSTR=6
NCORD=3
NFREE=3
C*** NSTR= NO OF STRAINS. NCORD= NO OF COORDINATES NFREE= NO OF DOF P
MAX=NG*NSTR
DO 10 I=1,N
X= CORD(I,N)
XI=(X+1.)/2.
WI=WEIGHT(I,N)
DO 10 J=1,N
Y= CORD(J,N)
ETA=(Y+1.)/2.
WJ= WEIGHT(J,N)
DO 10 K =1,N
Z= CORD(K,N)
ZI=(Z+1.0)/2.0
WK=WEIGHT(K,N)
CALL DERIVE( XI,ETA, ZI, R)
II=N*N*(I-1)+N*(J-1)+K
W(II)=WI*WJ*WK/8.0
DO 20 IJ=1,NS
IN=NS*(IJ-1)+IJ
DNX(IN)=R(1,IJ)
DNE(IN)=R(2,IJ)
DNZ(IN) =R(3,IJ)

```

```

20    CONTINUE
10    CONTINUE
C*****    NOW GENERATE THE MASTER VECTOR OF THE COORDINATES
      DO 30 J=1,NG
        NT=NS*(J-1)+1
        XX(NT;NS)=XE(1,1;NS)
        YY(NT;NS)=XE(1,2;NS)
        ZZ(NT;NS) = XE(1,3;NS)
30    CONTINUE
      LN=NG*NS
      DAJ11(1;LN)=DNX(1;LN)*XX(1;LN)
      DAJ12(1;LN)=DNX(1;LN)*YY(1;LN)
      DAJ13(1;LN) = DNX(1;LN)* ZZ(1;LN)
      DAJ21(1;LN)=DNE(1;LN)*XX(1;LN)
      DAJ22(1;LN)=DNE(1;LN)*YY(1;LN)
      DAJ23(1;LN) = DNE(1;LN)* ZZ(1;LN)
      DAJ31(1;LN) = DNZ(1;LN)* XX(1;LN)
      DAJ32(1;LN) = DNZ(1;LN)* YY(1;LN)
      DAJ33(1;LN) = DNZ(1;LN)* ZZ(1;LN)
      DO 40 I=1,NG
        NT=NS*(I-1)+1
        AJ11(I)= Q8SSUM(DAJ11(NT;NS))
        AJ12(I)= Q8SSUM(DAJ12(NT;NS))
        AJ13(I) =Q8SSUM(DAJ13(NT;NS))
        AJ21(I)= Q8SSUM(DAJ21(NT;NS))
        AJ22(I)= Q8SSUM(DAJ22(NT;NS))
        AJ23(I) =Q8SSUM(DAJ23(NT;NS))
        AJ31(I) =Q8SSUM(DAJ31(NT;NS))
        AJ32(I) =Q8SSUM(DAJ32(NT;NS))
        AJ33(I) =Q8SSUM(DAJ33(NT;NS))
40    CONTINUE
      DET11(1;NG) =AJ22(1;NG)*AJ33(1;NG)-AJ32(1;NG)*AJ23(1;NG)
      DET12(1;NG) =AJ21(1;NG)*AJ33(1;NG)-AJ31(1;NG)*AJ23(1;NG)
      DET13(1;NG) =AJ21(1;NG)*AJ32(1;NG)-AJ31(1;NG)*AJ22(1;NG)
      DET21(1;NG) =AJ12(1;NG)*AJ33(1;NG)-AJ32(1;NG)*AJ13(1;NG)
      DET22(1;NG) =AJ11(1;NG)*AJ33(1;NG)-AJ31(1;NG)*AJ13(1;NG)
      DET23(1;NG) =AJ11(1;NG)*AJ32(1;NG)-AJ31(1;NG)*AJ12(1;NG)
      DET31(1;NG) =AJ12(1;NG)*AJ23(1;NG)-AJ22(1;NG)*AJ13(1;NG)
      DET32(1;NG) =AJ11(1;NG)*AJ23(1;NG)-AJ21(1;NG)*AJ13(1;NG)
      DET33(1;NG) =AJ11(1;NG)*AJ22(1;NG)-AJ21(1;NG)*AJ12(1;NG)
      DET(1;NG) =AJ11(1;NG)*DET11(1;NG)-AJ12(1;NG)*DET12(1;NG)+
      ZAJ13(1;NG)* DET13(1;NG)
      AJI11(1; NG)  = DET11(1;NG)/DET(1;NG)
      AJI12(1; NG)  =- DET21(1;NG)/DET(1;NG)
      AJI13(1; NG)  = DET31(1;NG)/DET(1;NG)
      AJI21(1; NG)  =-DET12 (1;NG)/DET(1;NG)
      AJI22(1; NG)  = DET22(1;NG)/DET(1;NG)
      AJI23(1; NG)  =- DET32(1;NG)/DET(1;NG)
      AJI31(1; NG)  = DET13(1;NG)/DET(1;NG)
      AJI32(1; NG)  =- DET23(1;NG)/DET(1;NG)
      AJI33(1; NG)  = DET33(1;NG)/DET(1;NG)
C***    JOACOBIANS AND THEIR INVERSES ARE READY
      DO 50 J=1,NG
        NT=NS*(J-1)+1
        DSX(NT;NS) =DNX(NT;NS)*AJI11(J)+DNE(NT;NS)*AJI12(J) +DNZ(NT;NS)*
        Z AJI13(J)
        DSY(NT;NS) =DNX(NT;NS)*AJI21(J) +DNE(NT;NS)*AJI22(J) +DNZ(NT;NS)*
        Z AJI23(J)
        DSZ(NT;NS) =DNX(NT;NS)*AJI31(J) +DNE(NT;NS)*AJI32(J) +DNZ(NT;NS)*
        Z AJI33(J)
50    CONTINUE
C*****    CARTESIAN DERIVATIVES ARE READY
      IF(ICODE.EQ.2) GO TO 320
      I1=1
      I2=NS
      CALL Q8INTVAL (0,0,I1,0,I2,0,INVA(1;NG))

```



```

        LN= NS*NG
        DO 60 I=1,NS
        NT= NG*(I-1)+1
        DNX(NT;NG)=Q8VGATHR(DSX(I;LN),INVA(1;NG);DNX(NT;NG))
        DNE(NT;NG)=Q8VGATHR(DSY(I;LN),INVA(1;NG);DNE(NT;NG))
        DNZ(NT;NG)=Q8VGATHR(DSZ(I;LN),INVA(1;NG);DNZ(NT;NG))
60      CONTINUE
C****  CARTESIAN DERIVATIVES ARE REORDERED SO THAT THE DERIVATIVES AT A
C***   GAUSSIAN POINTS ARE GROUPED
        LEN=NS*NG
        BJ(1,1;LEN)= DNX(1;LEN)
        BJ(1,2;LEN)= DNE(1;LEN)
        BJ(1,3;LEN)= DNZ(1;LEN)
C***** COMPUTE THE PRODUCT OF WEIGHT AND DETRMINENTS
        WDUM(1;NG)= DET(1;NG)*W(1;NG)
        RETURN
320     CONTINUE
        LEN=NS*NG
        BJ(1,1;LEN)= DSX(1;LEN)
        BJ(1,2;LEN)= DSY(1;LEN)
        BJ(1,3;LEN)= DSZ(1;LEN)
        WDUM(1;NG)= DET(1;NG)*W(1;NG)
        RETURN
END
SUBROUTINE STRESS(DIS,XE,D,STR,STRS,B,WDUM)
C
C
COMMON/D382/NNPE,NDF,NQD,NSTR,NQD2,NNPE2,NQD2NPE,NQD2SR,MXQ2S
DIMENSION DIS(8,3), XE(8,3),D(6,6)
DIMENSION STRS(8,6),STR(8,6)
DIMENSION WDUM( 8), SUM(64), B( 64,3), DISP(64,3)
DIMENSION IREPL(20),STRD(6,8)
DESCRIPTOR IREPLD,SORCD,DESTD
C
C
DIMENSION IBSP(6,3)
DATA IBSP / 1,2*0, 2, 0, 3,
Z          0, 2, 0, 1, 3, 0,
Z          2*0, 3, 0, 2, 1 /
C
C
DATA NS, NSTR, NSH, NFREE / 8, 6, 3, 3 /
C***** NS= NUMBER OF SHAPE FUNCTIONS OR NODES ON THE ELEMENT
C***** NSTR= NUMBER OF STRAINS
C***** NSH= NUMBER OF INDEPENDENT DERIVATIVES IN THE B MATRIX
C***** NFREE= NUMBER OF DEGREES OF FREEDOM PER NODE
C
C
        LEN= NSTR*NQD2
        LDISP=NQD2NPE*NSH
        IREPL(1;NQD2)=0
        STRS(1,1;LEN)=0.0
        CALL ZEROLV(DISP,LDISP)
C
C****  PICK UP THE U,V,W DISPLACEMENTS SEPARATELY
C
C****  REPLICATE THE DISPLACEMENTS NQD2 TIMES
C
        ASSIGN IREPLD,IREPL(1;NQD2)
        DO 25 KC=1,NFREE
        ASSIGN SORCD, DIS(1,KC;NS)
        ASSIGN DESTD, DISP(1,KC;NQD2NPE)
        CALL Q8VXTOV (X'02', 0, IREPLD, 0, SORCD, 0, DESTD)
25      CONTINUE
C****  THE MASTER DISP VECTOR READY
C
C****  GET THE CARTESIAN DERIVATIVES AT THE NODES

```

```

      NJ=(J-1)*NQD2+1
220  FORC(IJ)=Q8SSUM(SUM(NJ;NQD2))
400  CONTINUE
      RETURN
      END
      SUBROUTINE MATINV(A,NMAX,N,B,MAX,M,DETERM)
      DIMENSION A(NMAX,M),B(NMAX,M)
      DIMENSION IPIVOT(100),INDEX(100,2),PIVOT(100)
C
C      IF M=0 IT CALCULATES THE INVERSE ONLY.
C      IF M=1 IT CALCULATES THE SOL TO AX=B IN B
C      INITIALIZATION
C
10    DETERM=1.0
15    DO 20 J=1,N
20    IPIVOT(J)=0
30    DO 550 I=1,N
C
C      SEARCH FOR THE PIVOT ELEMENT
C
40    AMAX=0.0
45    DO 105 J=1,N
50    IF(IPIVOT(J)-1)60,105,60
60    DO 100 K=1,N
70    IF(IPIVOT(K)-1)80,100,740
80    IF(ABS(AMAX)-ABS(A(J,K)))85,100,100
85    IROW= J
90    ICOLUM=K
95    AMAX= A(J,K)
100   CONTINUE
105   CONTINUE
110   IPIVOT(ICOLUM)=IPIVOT(ICOLUM)+1
C
C      INTERCHANGE ROWS TO PUT ELELMENG ON DIAGONAL
C
130   IF(IROW-ICOLUM)140,260,140
140   DETERM= -DETERM
150   DO 200 L=1,N
160   SWAP= A(IROW,L)
170   A(IROW,L)=A(ICOLUM,L)
200   A(ICOLUM,L)= SWAP
205   IF(M)260,260,210
210   DO 250 L=1,M
220   SWAP= B(IROW,L)
230   B(IROW,L)= B(ICOLUM,L)
250   B(ICOLUM,L)= SWAP
260   INDEX(I,1)= IROW
270   INDEX(I,2)= ICOLUM
310   PIVOT(I)= A(ICOLUM,ICOLUM)
320   DETERM= DETERM*PIVOT(I)
C
C      DIVIDE PIVOT BY PIVOT ELEMENT
C
330   A(ICOLUM,ICOLUM)=1.0
340   DO 350 L=1,N
350   A(ICOLUM,L)= A(ICOLUM,L)/PIVOT(I)
355   IF(M) 380,380,360
360   DO 370 L=1,M
370   B(ICOLUM,L)= B(ICOLUM,L)/PIVOT(I)
C
C      REDUCE NON-PIVOT ROWS
C
380   DO 550 L1=1,N
390   IF(L1-ICOLUM)400,550,400
400   T= A(L1,ICOLUM)
420   A(L1,ICOLUM)=0.0

```

ORIGINAL PAGE IS  
OF POOR QUALITY

```

LN= NS*NG
DO 60 I=1,NS
NT= NG*(I-1)+1
DNX(NT;NG)=Q8VGATHR(DSX(I;LN),INVA(1;NG);DNX(NT;NG))
DNE(NT;NG)=Q8VGATHR(DSY(I;LN),INVA(1;NG);DNE(NT;NG))
DNZ(NT;NG)=Q8VGATHR(DSZ(I;LN),INVA(1;NG);DNZ(NT;NG))
60    CONTINUE
C*****  CARTESIAN DERIVATIVES ARE REORDERED SO THAT THE DERIVATIVES AT A
C***    GAUSSIAN POINTS ARE GROUPED
LEN=NS*NG
BJ(1,1;LEN)= DNX(1;LEN)
BJ(1,2;LEN)= DNE(1;LEN)
BJ(1,3;LEN)= DNZ(1;LEN)
C*****  COMPUTE THE PRODUCT OF WEIGHT AND DETRMINENTS
WDUM(1;NG)= DET(1;NG)*W(1;NG)
RETURN
320    CONTINUE
LEN=NS*NG
BJ(1,1;LEN)= DSX(1;LEN)
BJ(1,2;LEN)= DSY(1;LEN)
BJ(1,3;LEN)= DSZ(1;LEN)
WDUM(1;NG)= DET(1;NG)*W(1;NG)
RETURN
END
SUBROUTINE STRESS(DIS,XE,D,STR,STRS,B,WDUM)
C
C
COMMON/D382/NNPE,NDF,NQD,NSTR,NQD2,NNPE2,NQD2NPE,NQD2SR,MXQ2S
DIMENSION DIS(8,3), XE(8,3),D(6,6)
DIMENSION STRS(8,6),STR(8,6)
DIMENSION WDUM( 8), SUM(64), B( 64,3), DISP(64,3)
DIMENSION IREPL(20),STRD(6,8)
DESCRIPTOR IREPLD,SORCD,DESTD
C
C
DIMENSION IBSP(6,3)
DATA IBSP / 1,2*0, 2, 0, 3,
Z          0, 2, 0, 1, 3, 0,
Z          2*0, 3, 0, 2, 1 /
C
C
DATA NS, NSTR, NSH, NFREE / 8, 6, 3, 3 /
C*****  NS= NUMBER OF SHAPE FUNCTIONS OR NODES ON THE ELEMENT
C*****  NSTR= NUMBER OF STRAINS
C*****  NSH= NUMBER OF INDEPENDENT DERIVATIVES IN THE B MATRIX
C*****  NFREE= NUMBER OF DEGREES OF FREEDOM PER NODE
C
C
LEN= NSTR*NQD2
LDISP=NQD2NPE*NSH
IREPL(1;NQD2)=0
STRS(1,1;LEN)=0.0
CALL ZEROLV(DISP,LDISP)
C
C*****  PICK UP THE U,V,W DISPLACEMENTS SEPARATELY
C
C*****  REPLICATE THE DISPLACEMENTS NQD2 TIMES
C
C
ASSIGN IREPLD,IREPL(1;NQD2)
DO 25 KC=1,NFREE
ASSIGN SORCD, DIS(1,KC;NS)
ASSIGN DESTD, DISP(1,KC;NQD2NPE)
CALL Q8VXTOV (X'02', 0, IREPLD, 0, SORCD, 0, DESTD)
25    CONTINUE
C*****  THE MASTER DISP VECTOR READY
C
C*****  GET THE CARTESIAN DERIVATIVES AT THE NODES

```

```

      CALL CDER( XE,NQD, B, WDUM, 2)
C
C*****      NOW DO THE PRODUCT D * B* DISPLACEMENTS
      DO 100 I=1,NSTR
      SUM(1;NQD2NPE)=0.0
      DO 110 J=1,NSH
      IT= IBSP(I,J)
      IF(IT.EQ.0) GO TO 110
      SUM(1;NQD2NPE)= SUM(1;NQD2NPE)+B(1,IT;NQD2NPE)* DISP(1,J;NQL PE)
110      CONTINUE
      DO 120 J=1,NQD2
      I1=(J-1)*NS+1
      STR(J,I)= Q8SSUM(SUM(I1;NS))
      STRD(I,J)=STR(J,I)
120      CONTINUE
100      CONTINUE
C
      DO 130 I=1,NQD2
      DO 140 J=1,NSTR
      STRS(I,J)= Q8SDOT (D(1,J;NSTR),STRD(1,I;NSTR))
140      CONTINUE
130      CONTINUE
      RETURN
      END
      SUBROUTINE FORCEP(BB,WTDET,STRS,FORC)
      COMMON/D382/NNPE,NDF,NQD,NSTR,NQD2,NNPE2,NQD2NPE,NQD2SR,MXQ2
      DIMENSION IBSP(6,3),B(64,3),WTDET(8),STRS(8,6)
      DIMENSION WTDETEX(64),SUM(64),SIG(64,6),IREPL(20)
      DIMENSION FORC(24),BB(64,3),INDX(8),SH(8)
      DATA IBSP / 1,2*0, 2, 0, 3,
Z          0, 2, 0, 1, 3, 0,
Z          2*0, 3, 0, 2, 1 /
C
C      NQD2=NQD*NQD*NQD
C      NQD2NPE=NQD2*NNPE
      DESCRIPTOR IREPLD, SORCD, DESTD
      DESCRIPTOR BDESC
      IREPL(1;NNPE)=0
      ASSIGN IREPLD, IREPL(1; NNPE)
      ASSIGN SORCD, WTDET(1; NQD2)
      ASSIGN DESTD, WTDETEX(1; NQD2NPE)
      CALL Q8VXTOV(X'02', 0, IREPLD, 0, SORCD, 0, DESTD)
      DO 155 IT=1,NDF
      DO 156 J=1,NNPE
      DO 150 II=1,NQD2
150      INDX(II)=(II-1)*NNPE+J
      SH(1;NQD2)=Q8VGATHR(BB(1,IT;NQD2NPE),INDX(1;NQD2);SH(1;NQD2))
      I1=(J-1)*NQD2+1
      B(I1,IT;NQD2)=SH(1;NQD2)
156      CONTINUE
      BB(1,IT;NQD2NPE) = B(1,IT;NQD2NPE)*WTDETEX(1;NQD2NPE)
155      CONTINUE
      DO 205 IS=1,NSTR
      DO 205 II=1,NNPE
      NQ=(II-1)*NQD2+1
205      SIG(NQ,IS;NQD2)=STRS(1,IS;NQD2)
      DO 400 II=1,NDF
      SUM(1;NQD2NPE)=0.
      DO 210 JJ=1,NSTR
      IT=IBSP(JJ,II)
213      IF (IT.EQ.0) GO TO 210
      ASSIGN BDESC ,BB(1,IT;NQD2NPE)
      SUM(1;NQD2NPE)=SUM(1;NQD2NPE)+BDESC*SIG(1,JJ;NQD2NPE)
210      CONTINUE
      DO 220 J=1,NNPE
      IJ=(J-1)*3+II

```

```

      NJ=(J-1)*NQD2+1
220  FORC(IJ)=Q8SSUM(SUM(NJ;NQD2))
400  CONTINUE
      RETURN
      END
      SUBROUTINE MATINV(A,NMAX,N,B,MAX,M,DETERM)
      DIMENSION A(NMAX,NMAX),B(NMAX,MAX)
      DIMENSION IPIVOT(100),INDEX(100,2),PIVOT(100)
C
C      IF M=0 IT CALCULATES THE INVERSE ONLY.
C      IF M=1 IT CALCULATES THE SOL TO AX=B IN B
C      INITIALIZATION
C
10    DETERM=1.0
15    DO 20 J=1,N
20    IPIVOT(J)=0
30    DO 550 I=1,N
C
C      SEARCH FOR THE PIVOT ELEMENT
C
40    AMAX=0.0
45    DO 105 J=1,N
50    IF(IPIVOT(J)-1)60,105,60
60    DO 100 K=1,N
70    IF(IPIVOT(K)-1)80,100,740
80    IF( ABS(AMAX)- ABS(A(J,K)))85,100,100
85    IROW= J
90    ICOLUM=K
95    AMAX= A(J,K)
100   CONTINUE
105   CONTINUE
110   IPIVOT(ICOLUM)=IPIVOT(ICOLUM)+1
C
C      INTERCHANGE ROWS TO PUT ELELMENG ON DIAGONAL
C
130   IF(IROW-ICOLUM)140,260,140
140   DETERM= -DETERM
150   DO 200 L=1,N
160   SWAP= A(IROW,L)
170   A(IROW,L)=A(ICOLUM,L)
200   A(ICOLUM,L)= SWAP
205   IF(M)260,260,210
210   DO 250 L=1,M
220   SWAP= B(IROW,L)
230   B(IROW,L)= B(ICOLUM,L)
250   B(ICOLUM,L)= SWAP
260   INDEX(I,1)= IROW
270   INDEX(I,2)= ICOLUM
310   PIVOT(I)= A(ICOLUM,ICOLUM)
320   DETERM= DETERM*PIVOT(I)
C
C      DIVIDE PIVOT BY PIVOT ELEMENT
C
330   A(ICOLUM,ICOLUM)=1.0
340   DO 350 L=1,N
350   A(ICOLUM,L)= A(ICOLUM,L)/PIVOT(I)
355   IF(M) 380,380,360
360   DO 370 L=1,M
370   B(ICOLUM,L)= B(ICOLUM,L)/PIVOT(I)
C
C      REDUCE NON-PIVOT ROWS
C
380   DO 550 L1=1,N
390   IF(L1-ICOLUM)400,550,400
400   T= A(L1,ICOLUM)
420   A(L1,ICOLUM)=0.0

```

```

430 DO 450 L=1,N
450 A(L1,L)= A(L1,L)-A(ICOLUM,L)*T
455 IF(M) 550,550,460
460 DO 500 L=1,M
500 B(L1,L)= B(L1,L)-B(ICOLUM,L)*T
550 CONTINUE

```

```

C
C INTERCHANGE COLUMNS
C

```

```

600 DO 710 I=1,N
610 L=N+1-I
620 IF(INDEX(L,1)-INDEX(L,2))630,710,630
630 JROW= INDEX(L,1)
640 JCOLUM= INDEX(L,2)
650 DO 705 K=1,N
660 SWAP= A(K,JROW)
670 A(K,JROW)= A(K,JCOLUM)
700 A(K,JCOLUM)= SWAP
705 CONTINUE
710 CONTINUE
740 RETURN
END

```

BLOCK DATA

COMMON/GAUSS/ CORD(8,8),WEIGHT(8,8)

COMMON/GENRL/GCR(8,3)

DATA CORD/ 8\*0.0,

```

A -0.577350269189626,0.577350269189626,6*0.0,
B -0.774596669241483,0.0,0.774596669241483,5*0.0,
C -0.861136311594053,-.339981043584856,0.339981043584856,
1 0.861136311594053,4*0.0,
D-0.906179845938664,-0.538469310105683,0.0,0.538469310105683,
1 0.906179845938664,3*0.0,
E -0.932469514203152,-0.661209386466265,-0.238619186083197,
1 +0.238619186083197,0.661209386466265,0.932469514203152,2*0.0,
F -0.949107912342759, -0.741531185599394,-0.405845151377397,0.0,
1 0.405845151377397, 0.741531185599394,0.949107912342759,0.0,
G -0.960289856497536,-0.796666477413627,-0.525532409916329,
1 -0.183434642495650,0.183434642495650,0.525532409916329,
2 0.796666477413627,0.960289856497536/

```

DATA WEIGHT /8\*0.0,

```

A 1.0,1.0, 6*0.0,
B 0.555555555555556,0.888888888888889,0.555555555555556,5*0.0,
C 0.347854845137454,0.652145154862546,0.652145154862546,
1 0.347854845137454,4*0.0,
D 0.236926885056189, 0.478628670499366,0.568888888888889,
1 0.478628670499366, 0.236926885056189,3*0.0,
E 0.171324492379170,0.360761573048139,0.467913934572691,
1 0.467913934572691,0.360761573048139,0.171324492379170,2*0.0 ,
F 0.129484966168870,0.279705391489277,0.381830050505119,
1 0.417959183673469,0.381830050505119,0.279705391489277,
2 0.129484966168870 ,0.0,
G 0.101228536290376,0.222381034453374,0.313706645877887,
1 0.362683783378362,0.362683783378362,0.313706645877887,
2 0.222381034453374,0.101228536290376/
DATA GCR/0.0,0.0,1.0,1.0,0.0,0.0,1.0,1.0,
1 1.0,0.0,0.0,1.0,1.0,0.0,0.0,1.0,
2 4*0.0,4*1.0/

```

END

SUBROUTINE VON(STR,SBAR,PFS)

```

C
C *** COMPUTE FLOW VECTOR
C

```

DIMENSION PFS(6),STR(6)

S1=2\*SBAR

PFS(1)=(2\*STR(1)-STR(2)-STR(3))/S1

```

PFS(2)=(2*STR(2)-STR(1)-STR(3))/S1
PFS(3)=(2*STR(3)-STR(1)-STR(2))/S1
PFS(4)=(6*STR(4))/S1
PFS(5)=(6*STR(5))/S1
PFS(6)=(6*STR(6))/S1
RETURN
END
SUBROUTINE DEPL(STR,SBAR,D,HP,DPL)
DIMENSION STR(6),STA(6),D(6,6),DD(6),DPL(6,6)
C
C *** DEP - ELASTIC-PLASTIC MATRIX
C
CALL VON(STR,SBAR,STA)
B1=D(1,1)
B2=D(4,4)
B3=D(1,2)
DD(1)=B1*STA(1)+B3*(STA(2)+STA(3))
DD(2)=B1*STA(2)+B3*(STA(1)+STA(3))
DD(3)=B1*STA(3)+B3*(STA(1)+STA(2))
DD(4)=B2*STA(4)
DD(5)=B2*STA(5)
DD(6)=B2*STA(6)
SD=B1*(STA(1)**2+STA(2)**2+STA(3)**2)+2*B3*(STA(1)*STA(2)+
C STA(2)*STA(3)+STA(3)*STA(1))+B2*(STA(4)**2+STA(5)**2+
C STA(6)**2)
SD=1.0/(SD+HP)
DO 10 I=1,6
DO 10 J=1,6
10 DPL(I,J)=D(I,J)-DD(I)*DD(J)*SD
RETURN
END
SUBROUTINE SEQU(XT,XXZ)
C
VON MISES YIELD CRITERION.
DIMENSION XT(6)
S1=0.5*(XT(1)-XT(2))**2
S2=0.5*(XT(2)-XT(3))**2
S3=0.5*(XT(3)-XT(1))**2
S4=3*(XT(4))**2
S5=3*(XT(5))**2
S6=3*(XT(6))**2
ST=S1+S2+S3+S4+S5+S6
XXZ=SQRT(ST)
RETURN
END
SUBROUTINE MULTYS(A,B,N,M,C)
DIMENSION A(N,M),B(M),C(N)
DO 10 I=1,N
C(I)=0.0
DO 10 J=1,M
10 C(I)=C(I)+A(I,J)*B(J)
RETURN
END
SUBROUTINE PLAS
COMMON/MAIN/AA(2000000),BB(9600,1),D(6,6),DINV(6,6),
1DISP(80),EPS(12400),EFEST(1550),FORCE(10),LINE(100),
2LOCAT(10),LBFOR(10),MB(9600),MSUM(9600),MPTAB(9600),
3MPLAS(12400),MODE(8,1550),MPLC(1550),NODXO(700),
4NODYO(700),NODZO(700),NODXC(700),NODYC(700),NODZC(700),
5NODFIX(80),NODLOD(80),NDISP(80),R(9600),
6SIGBAR(12400),SK(24,24),T1(9600),T2(9600),T4(2000),
7T3(9600),U(3200),UOLD(3200),V(3200),VOLD(3200),V2(3200),
8W(3200),WOLD(3200),X(74400),XR(3200),Y(74400),
9YR(3200),Z(9600),ZR(3200)
COMMON/CNST/EPsi,SK2,LMAX,KMAX,DAX,
1PYLD,SCRIT,YOUNG,POIS,CRACK,PT,WIDTH,PMAX,HP,
2SBAR,LPRIT,NGAUS,NLAYER,HNODE,

```

```

3INODXO,INODZO,INODXC,INODYC,INODZC,LNSTIF,MXNOD,
4MXNEL,MXGAUS,ICUT,LTOTB,ITNODX,KLU,NTYP,NLM,
5NDOF,KNEW,NEP,ERIT,NELM,AN,ROM
C
COMMON/MLTNMAT/YSTRS(20),YSTRN(20),PLMODR(20),NSEGMT
COMMON/D382/NNPE,NDF,NQD,NSTR,NQD2,NNPE2,NQD2NPE,NQD2SR,MXQ2S
COMMON/VECT/ STRV(8,6),STRSV(8,6),BMT(64,3),WDUM(8),XE(8,3),
C NCUBE(8),DIS(8,3)
DIMENSION DPL(6,6),AMAT(8,3),STGAS(6)
DIMENSION QP(9600),STGASV(8,6),FFTR(40)
DIMENSION STR(6),U2(3200),INDX(3200)
DIMENSION YTP(6),ST1(6),DPSTRN(6),STREP(6)
DIMENSION PLV(24)
DATA HALT,GROW/4HHALT,4HGROW/
1000 FORMAT(5X,'STEP#',I2,2X,'TIME:CPU=',F12.4,2X,
1 'WALL=',F12.3)
C
C *** INCREMENT DISPLACEMENTS, FORCES, STRESS & STRAINS TO 1ST YIELD LOAD
C
UOLD(1;NNODE)=U(1;NNODE)*PYLD
VOLD(1;NNODE)=V(1;NNODE)*PYLD
WOLD(1;NNODE)=W(1;NNODE)*PYLD
R(1;NDOF)=R(1;NDOF)*PYLD
X(1;MXQ2S)=X(1;MXQ2S)*PYLD
Y(1;MXQ2S)=Y(1;MXQ2S)*PYLD
C *** ZEROING
QP(1;NDOF)=0.0
EPS(1;MXGAUS)=0.0
MPLAS(1;MXGAUS)=0
C
C *** READ DATA
C
READ(5,11) PCT,ERIT,MAXIT,NODE1,NODE2,NELE1,NELE2
11 FORMAT(2E10.3,5I5)
READ(5,312) P,WORD
312 FORMAT(E10.3,1X,A4)
WRITE(6,313) P,PCT,ERIT,MAXIT,NODE1,NODE2,NELE1,NELE2
313 FORMAT(///10X,'TOTAL LOAD FACTOR=',F10.4
1/10X,'INCREMENTAL LOAD FACTOR=',F10.4/10X,'ALLOWABLE ERROR ON STRE
2SS=',F10.4/10X,'MAXIMUM NUMBER OF ITERATION=',I4
3/10X,'PRINT DISPLACEMENTS AT NODES',I5,' TO',I5
4/10X,'PRINT STRESSES IN ELEMENTS',I5,' TO',I5)
CALL Q3CLOCKS(CPU,WALL)
IZIP1=0
WRITE(6,1000) IZIP1,CPU,WALL
PT=PYLD
CALL PLOUT(NODE1,NODE2,NELE1,NELE2)
CALL Q3CLOCKS(CPU,WALL)
IZIP1=1
WRITE(6,1000) IZIP1,CPU,WALL
IF(NEP.EQ.0) STOP
DELP=PCT*PYLD
NPL=0
20 PMAX=PT
PT=PT+DELP
NPL=0
IF(PT.GE.P.AND.DELP.GT.0.0) GO TO 25
IF(PT.LE.P.AND.DELP.LT.0.0) GO TO 25
GO TO 26
25 PT=P
NPL=1
26 DELPO=PT-PMAX
NPL=NPL+1
KLU=0
ICON=0
V2(1;NNODE)=VOLD(1;NNODE)
PLAS
PLAS

```



```

C
C   HOLDING APPLIED LOAD CONSTANT -ITERATE UNTIL SOLUTION CONVERGE
C
   PTOY=PT/PYLD
   NBREAK=0
   GO TO 45
35  NL=-1
36  NL=NL+1
   NPLOT=0
   IF(NL.GT.NLM) GO TO 91
   ANL=NL
   DO 133 JIS=1,ICUT
   FFTR(JIS)=FORCE(JIS)*(1.-ANL/NLM)
133  WRITE(6,167)FFTR(JIS),NL
167  FORMAT(10X,"CRACK-TIP FORCE=",E16.7,"AT STEP",I2)
45  DO 50 ITER=1,MAXIT
   MC=0
65  BB(1,1;NDOF)=QP(1;NDOF)+R(1;NDOF)*PTOY
   IF(KLU.EQ.1) GO TO 830
   GO TO 831
830  DO 134 JIT=1,ICUT
   NOM=LOCAT(JIT)
   NFL=3*NOM-1
134  BB(NFL,1)=BB(NFL,1)+FFTR(JIT)
831  CONTINUE
   IFAC=1
   NC=1
   CALL SYMBAN(LNSTIF,NDOF,NB,MSUM,AA,1,BB,IFAC,T1,IERR,
1  ALP,Z,T2,T3,T4,NC)
   DO 70 N=1,NNODE
70  INDX(N)=(N-1)*3+1
   U2(1;NNODE)= Q8VGATHR(BB(1,1;NDOF),INDX(1;NNODE);U2(1;NNODE))
   U(1;NNODE)=U2(1;NNODE)-UOLD(1;NNODE)
   UOLD(1;NNODE)=U2(1;NNODE)
   INDX(1;NNODE)=INDX(1;NNODE)+1
   U2(1;NNODE)= Q8VGATHR(BB(1,1;NDOF),INDX(1;NNODE);U2(1;NNODE))
   V(1;NNODE)=U2(1;NNODE)-VOLD(1;NNODE)
   VOLD(1;NNODE)=U2(1;NNODE)
   INDX(1;NNODE)=INDX(1;NNODE)+1
   U2(1;NNODE)= Q8VGATHR(BB(1,1;NDOF),INDX(1;NNODE);U2(1;NNODE))
   W(1;NNODE)=U2(1;NNODE)-WOLD(1;NNODE)
   WOLD(1;NNODE)=U2(1;NNODE)

C
C   COMPUTE TOTAL STRAIN INCREMENTS FROM DISPLACEMENT INCREMENTS.
C   COMPUTE ELASTIC STRESS INCREMENTS AND ADD TO CURRENT STRESSES.
C   CHECK YIELD CONDITION FOR PLASTIC ELEMENTS.
C
   IGAUSP=0
   DO 80 I=1,NELM
   DO 75 J=1,8
   NCUBE(J)=MODE(J,I)
   N1=NCUBE(J)
   DIS(J,1)=U(N1)
   DIS(J,2)=V(N1)
75  DIS(J,3)=W(N1)
C ***
   CALL CORDIN(NCUBE,MXNOD,XR,YR,ZR,XE)
   CALL STRESS(DIS,XE,D,STRV,STRSV,BMT,WDUM)
   STGASV(1,1;NQD2SR)=0.0
   ILOC=(I-1)*NQD2SR
   DO 76 IG=1,NQD2
   IGAUSP=IGAUSP+1
   SBAR=SIGBAR(IGAUSP)
   DO 77 JS=1,6
   STR(JS)=STRSV(IG,JS)
   YTR(JS)=STRV(IG,JS)

```

```

JS1=ILOC+IG+NQD2*(JS-1)
77 ST1(JS)=X(JS1)
   CALL SEQU(ST1,S1)
   A11=STR(1)
   A22=STR(2)
   A33=STR(3)
   A44=STR(4)
   A55=STR(5)
   A66=STR(6)
   S11=ST1(1)
   S22=ST1(2)
   S33=ST1(3)
   S44=ST1(4)
   S55=ST1(5)
   S66=ST1(6)
   ST1(1;6)=ST1(1;6)+STR(1;6)
   CALL SEQU(ST1,S2)
C *** IF SOL CONVERGED THEN GOTO 801
   IF(ICON.EQ.1) GO TO 801
   IF(S2.LT.S1) MPLAS(IGAUSP)=0
   IF(S2.LT.S1) GO TO 801
   IF(MPLAS(IGAUSP).NE.O.AND.ITER.GT.1) GO TO 74
   IF(S2.LE.SBAR) GO TO 801
   MPLAS(IGAUSP)=IGAUSP
74 MP=1
C CHECK FOR CONVERGENCE.
   IF(ABS(S2-SBAR).GT.ERIT) MC=1
   A=A11**2+A22**2+A33**2+3*(A44**2)+3*(A55**2)+3*(A66**2)
1 -(A11*A22)-(A22*A33)-(A33*A11)
   B22=S11*(2*A11-A22-A33)+S22*(2*A22-A11-A33)+S33*(2*A33-A22-A11)
1 +6*S44*A44+6*S55*A55+6*S66*A66
   C=S1**2-SBAR**2
   IF(A.LT.EPSI) GO TO 8
   IF(ITER.EQ.2) GO TO 200
   DELTA=B22**2-4*A*C
200 IF(DELTA)200,40,40
   PX=(SBAR-S1)/(S2-S1)
   GO TO 231
40 PONE=(-B22+SQRT(DELTA))/(2.*A)
   PTWO=(-B22-SQRT(DELTA))/(2.*A)
   PX=PONE
   IF(ABS(PCWE).GT.ABS(PTWO))PX=PTWO
231 CONTINUE
   PXD=1.-PX
   YTR(1;6)=YTR(1;6)*PXD
   STR(1;6)=STR(1;6)*PXD
   IF(ROM.LE.O. .AND. NSEGIT.EQ.O) HP=AM*YOUNG
   IF(ROM.LE.O. .AND. NSEGMT.GT.O) HP=FNMAT(SBAR,YOUNG,EPS(IGAUSP),
CIGAUSP)
   IF(ROM.GT.O.) HP=ROM**AM*SBAR**(1.-AM)/AM
   CALL DEPL(ST1,SBAR,D,HP,DPL)
   CALL MULTYS(DPL,YTR,6,6,STREP)
   CALL MULTYS(DINV,STREP,6,6,DPSTRN)
   DPSTRN(1;6)=YTR(1;6)-DPSTRN(1;6)
   STGAS(1;6)=STR(1;6)-STREP(1;6)
   CALL ERTA(DPSTRN,SMA)
   EPS(IGAUSP)=EPS(IGAUSP)+SMA
   HC=1.0
   SIGBAR(IGAUSP)=SBAR+HC*HP*SMA
   DO 12 IP=1,6
12 STGASV(IG,IP)=STGAS(IP)
8 CONTINUE
C IF(1.EQ.1) WRITE(6,400) SBAR,EPS(IGAUSP),SIGBAR(IGAUSP),HP,S1,S2
C400 FORMAT(5X,'CONSTANTS: ',6F12.3)
76 CONTINUE
   STRSV(1,1;NQD2SR)=STRSV(1,1;NQD2SR)-STGASV(1,1;NQD2SR)

```

```

CALL FORCEP(BMT,WDUM,STGASV,PLV)
DO 455 IT=1,8
  IX=3*IT-2
  N1=NCUBE(IT)
  NU=3*N1-2
  QP(NU) =QP(NU)+PLV(IX)
  QP(NU+1) =QP(NU+1)+PLV(IX+1)
455   QP(NU+2)=QP(NU+2)+PLV(IX+2)
801  ILOC1=ILOC+1
      X(ILOC1;NQD2SR)=X(ILOC1;NQD2SR)+STRSV(1,1;NQD2SR)
      Y(ILOC1;NQD2SR)=Y(ILOC1;NQD2SR)+STRV(1,1;NQD2SR)
80   CONTINUE
      IF(ICON.EQ.1) GO TO 90
      IF(MP.EQ.0) GO TO 90
      IF(MC.EQ.1) GO TO 49
      WRITE(6,52)ITER
52   FORMAT(10X,'SOLUTION CONVERGED IN',I4,'ITERATIONS')
      ICON=1
      GO TO 49
53   WRITE(6,54)ITER
54   FORMAT(10X,'NO CONVERGENCE IN ',I4,'ITERATION')
      CALL PLOUT(NODE1,NODE2,NELE1,NELE2)
      GO TO 999
49   IF(ITER.EQ.MAXIT) GO TO 53
      IF(KLU.EQ.1.AND.NL.EQ.0) GO TO 90
50   CONTINUE
90   CONTINUE
      MP=0
      ICON=0
      IF(KLU.EQ.1) GO TO 36
91   CONTINUE
      CALL CONTACT
      IF(KNEW.EQ.1) GO TO 45
      IF(NPLOT.EQ.1) CALL PLOUT(NODE1,NODE2,NELE1,NELE2)
      IF(NPL.EQ.NEP) CALL PLOUT(NODE1,NODE2,NELE1,NELE2)
      IF(NPL.EQ.NEP) NPL=0
      IF(NTYP.EQ.1) CALL BREAK
      IF(NTYP.EQ.1) GO TO 100
      IF(KLU.EQ.1) GO TO 100
      IF(NPLOT.EQ.1.AND.WORD.EQ.GROW) CALL BREAK
100  CONTINUE
      IF(KLU.EQ.2) GO TO 999
      IF(KLU.EQ.3) GO TO 999
      IF(KLU.EQ.1) GO TO 35
      IF(NPLOT.EQ.1) GO TO 99
      GO TO 20
99   CONTINUE
102  DELP=--DELP
      READ(5,121)P,WORD
121  FORMAT(E10.3,1X,A4)
      IF(WORD.EQ.HALT) GO TO 999
      NPL=0
      MPLAS(1;MXGAUS)=0
      GO TO 20
999  RETURN
END
      SUBROUTINE ERTA(A,B)
      DIMENSION A(6)
      X=A(1)
      Y=A(2)
      Z=A(3)
      XY=A(4)/2.
      YZ=A(5)/2.
      ZX=A(6)/2.
      S1=(X-Y)**2
      S2=(Y-Z)**2

```

```

      S3=(Z-X)**2
      S4=6*(XY)**2
      S5=6*(YZ)**2
      S6=6*(ZX)**2
      STOT=S1+S2+S3+S4+S5+S6
      SPO=SQRT(STOT)
      SO=SQRT(2.)/3.
      B=SO*SPO
      RETURN
      END
      SUBROUTINE PLOUT(NODE1,NODE2,NELE1,NELE2)
      COMMON/MAIN/AA(2000000),BB(9600,1),D(6,6),DINV(6,6),
      1DISP(80),EPS(12400),EFEST(1550),FORCE(10),LINE(100),
      2LOCAT(10),LBFOR(10),MB(9600),MSUM(9600),MPTAB(9600),
      3MPLAS(12400),MODE(8,1550),MPLC(1550),NODXO(700),
      4NODYO(700),NODZO(700),NODXC(700),NODYC(700),NODZC(700),
      5NODFIX(80),NODLOD(80),NDISP(80),R(9600),
      6SIGBAR(12400),SK(24,24),T1(9600),T2(9600),T4(2000),
      7T3(9600),U(3200),UOLD(3200),V(3200),VOLD(3200),V2(3200),
      8W(3200),WOLD(3200),X(74400),XR(3200),Y(74400),
      9YR(3200),Z(9600),ZR(3200)
      COMMON/CNST/EPSI,SK2,LMAX,KMAX,DAX,
      1PYLD,SCRIT,YOUNG,POIS,CRACK,PT,WIDTH,PMAX,HP,
      2SBAR,LPRIT,NGAUS,NLAYER,NNODE,
      3INODXO,INODYO,INODZO,INODXC,INODYC,INODZC,LNSTIF,MXNOD,
      4MXNEL,MXGAUS,ICUT,LTOTB,ITNODX,KLU,NTYP,NLM,
      5NDOF,KNEW,NEP,ERIT,NELM,AM,ROM

      COMMON/D382/NNPE,NDF,NQD,NSTR,NQD2,NNPE2,NQD2NPE,NQD2SR,MXQ2S
      COMMON/VECT/ STRV(8,6),STRSV(8,6),BMT(64,3),WDUM(8),XE(8,3),
      C NCUBE(8),DIS(8,3)
      DIMENSION STRS(6),FRC(24),FORCEX(3200),FORCEY(3200),FORCEZ(3200)

C
C *** OUTPUT ROUTINE
C
      WRITE(6,10)PT,CRACK,WIDTH
10    FORMAT(/,10X,"APPLIED LOAD=",E12.5,8X,"CRACK=",
1    F10.5,10X,"WIDTH=",F10.5/)
      IF(CRACK.LT.EPSI) GO TO 20
      WRITE(6,15)
15    FORMAT(12X,"NODE",5X,"X",10X,"Y",10X,"Z",10X,
1    "COD"/)
      CRACK1=CRACK+EPSI
      CRACK2=CRACK-10*DAX
      DO 16 I=1,INODYO
      L=NODYO(I)
      IF(XR(L).LT.CRACK2) GO TO 16
      IF(YR(L).GT.EPSI) GO TO 16
      IF(XR(L).GT.CRACK1) GO TO 16
      WRITE(6,25)L,XR(L),YR(L),ZR(L),VOLD(L)
25    FORMAT(5X,I4,4E14.6)
16    CONTINUE
20    CONTINUE
      WRITE(6,30)
30    FORMAT(/,30X,"DISPLACEMENTS"/,12X,"NODE",14X,"U",
1    18X,"V",18X,"W")
      DO 12 N=NODE1,NODE2
12    WRITE(6,22) N,UOLD(N),VOLD(N),WOLD(N)
22    FORMAT(10X,I5, 5X,3(E13.6, 3X))
      WRITE(6,35)
35    FORMAT(/,20X,"STRESSES AND STRAINS",10X,1H*,3X,
1    "DENOTES PLASTIC ELEMENTS",30X,"EFFECTIVE"/,5X,
1    "ELEMENT",5X,"SIGX",8X,"SIGY",8X,"SIGZ",8X,"TAUXY",
1    8X,"TAUZY",8X,"TAUZX",8X,"EFFE-STRESS"/)
C
C *** CALCULATE NODAL FORCES AND GAUSS POINT STRESSES

```

```

C
WRITE(6,25) NGAUS
NGAUS=2
FORCEX(1;NNODE)=0.0
FORCEY(1;NNODE)=0.0
FORCEZ(1;NNODE)=0.0
N=0
IGAUSP=0
DO 300 IE=1,NELM
  NGUBE(1;8)=MODE(1,IE;8)
  CALL CORLIN(NGUBE,MKNOD,XR,YR,Z,XE)
  ILOC1=(IE-1)*NQD2SR+1
  STRSV(1,1;NQD2SR)=X(ILOC1;NQD2SR)
  CALL CDER(XE,NGAUS,BMT,WDUM,2)
  CALL FORCEP(BMT,WDUM,STRSV,FR)
  DO 352 I=1,8
    I1=NCUBE(I)
    IX=I*3-2
    FORCEX(I1)=FORCEX(I1)+FRC(IX)
    FORCEY(I1)=FORCEY(I1)+FRC(IX)
352  FORCEZ(I1)=FORCEZ(I1)+FRC(IX)
    DO 350 IG=1,NQD2
      IGAUSP=IGAUSP+1
    DO 351 I=1,6
351  STRS(I)=STRSV(IG,I)
      SIGO=(1-ERIT)*SIGBAR(IGAUSP)
      CALL SEQU(STRS,STP)
      IF(STP.GE.SIGO) GO TO 42
      WRITE(6,43)IE,IG,(STRS(L),L= 6),STP
43  FORMAT(5X,I6, I2,5X,7E12.5)
      GO TO 350
42  WRITE(6,45)IE,IG,(STRS(L),L= 6),STP
45  FORMAT(4X,I6, I2,5X,7E12.5)
      KGAUSP=KGAUSP+1
350  CONTINUE
      IF(KGAUSP.LE.0) GOTO 300
93  N=N+1
      MPLC(N)=IE
300  CONTINUE
      NOPL=N
C
C ***
C
WRITE(6,371)
DO 370 IN=NODE1,NODE2
370  WRITE(6,372) IN,FORCEX(IN),FORCEY(IN),FORCEZ(IN)
371  FORMAT(1H1///10X,"NODE #",5X, "ORCEX",8X,"FORCEY",7X,"FORCEZ")
372  FORMAT(10X,I5,2X,3(E12.4,2X))
C
C ***
C
WRITE(6,380)
380  FORMAT(//10X,"LIST OF PLASTIC ELEMENTS//")
WRITE(6,381) (MPLC(I),I=1,N)
381  FORMAT(5X,20I5)
997  RETURN
END
SUBROUTINE CONTACT
COMMON/MAIN/AA(2000000),BB(96,1),D(6,6),DINV(6,6),
1DISP(80),EPS(12400),EFEST(155),FORCE(10),LINE(100),
2LOCAT(10),LBFOR(10),MB(9600),UM(9600),MPTAB(9600),
3MPLAS(12400),MODE(8,1550),MPL(1550),NODXO(700),
4NODYO(700),NODZO(700),NODXC(7),NODYC(700),NODZC(700),
5NODFIX(80),NODLOD(80),NDISP(8),R(9600),
6SIGBAR(12400),SK(24,24),T1(9600),T2(9600),T4(2000),
7T3(9600),U(3200),UOLD(3200),V(200),VOLD(3200),V2(3200),

```

```

      8W(3200),WOLD(3200),X(74400),XR(3200),Y(74400),
      9YR(3200),Z(9600),ZR(3200)
      COMMON/CNST/EPSI,SK2,LMAX,KMAX,DAX,
1    1PYLD,SCRIT,YOUNG,POIS,CRACK,PT,WIDTH,PMAX,HP,
      2SBAR,LPRIT,NGAUS,NLAYER,NNODE,
      3INODXO,INODYO,INODZO,INODXC,INODYC,INODZC,LNSTIF,MXNOD,
      4MXNEL,MXGAUS,ICUT,LTOTB,ITNODX,KLU,NTYP,NLM,
      5NDOF,KNEW,NEP,ERIT,NELM,AM,ROM

C      CHANGES SPRING STIFNESS IF CRACK CLOSES OR OPENS.
C      DONT FORGET TO PUT THE COMMON HERE.
      WRITE(6,95)
95     FORMAT(5X,"CALLING FROM THE CONTACT")
      DO 10 I=1,INODYO
      L=NODYO(I)
      KNEW=0
      CX=XR(L)-CRACK
      IF(CX.LT.-EPSI) GO TO 9
      GO TO 40
9      NV=3*L-1
      MPTX=MPTAB(NV)
      IF(VOLD(L).LE.0.0) MPTAB(NV)=1
      IF(VOLD(L).GT.0.0) MPTAB(NV)=0
      IF(MPTX.NE.MPTAB(NV)) KNEW=1
      IF(KNEW.EQ.0) GO TO 40
      PN=PT-((PT-PMAX)/(VOLD(L)-V2(L)))*VOLD(L)
      Z(NV)=1.0
      NC=NV
      ALP=SK2
      IF(MPTAB(NV).EQ.0) ALP=-SK2
      IFAC=3
      CALL SYMBAN(LNSTIF,NDOF,MB,MSUM,AA,1,BB,IFAC,T1,IERR,
1    ALP,Z,T2,T3,T4,NC)
      IF(MPTAB(NV).EQ.0) WRITE(6,20)L,PN
20     FORMAT(/,2X,"NODE",I3,"OPENED AT",F8.3)
      IF(MPTAB(NV).EQ.1) WRITE(6,30)L,PN
30     FORMAT(/,2X,"NODE",I3,"CLOSED AT",F8.3)
40     CONTINUE
10     CONTINUE
      RETURN
      END
      SUBROUTINE BREAK
      COMMON/MAIN/AA(2000000),BB(9600,1),D(6,6),DINV(6,6),
      1DISP(80),EPS(12400),EFEST(1550),FORCE(10),LINE(100),
      2LOCAT(10),LBFOR(10),MB(9600),MSUM(9600),MPTAB(9600),
      3MPLAS(12400),NODE(8,1550),MPLC(1550),NODXO(700),
      4NODYO(700),NODZO(700),NODXC(700),NODYC(700),NODZC(700),
      5NODFIX(80),NODLOD(80),NDISP(80),R(9600),
      6SIGBAR(12400),SK(24,24),T1(9600),T2(9600),T4(2000),
      7T3(9600),U(3200),UOLD(3200),V(3200),VOLD(3200),V2(3200),
      8W(3200),WOLD(3200),X(74400),XR(3200),Y(74400),
      9YR(3200),Z(9600),ZR(3200)
      COMMON/CNST/EPSI,SK2,LMAX,KMAX,DAX,
      1PYLD,SCRIT,YOUNG,POIS,CRACK,PT,WIDTH,PMAX,HP,
      2SBAR,LPRIT,NGAUS,NLAYER,NNODE,
      3INODXO,INODYO,INODZO,INODXC,INODYC,INODZC,LNSTIF,MXNOD,
      4MXNEL,MXGAUS,ICUT,LTOTB,ITNODX,KLU,NTYP,NLM,
      5NDOF,KNEW,NEP,ERIT,NELM,AM,ROM

C      COMMON GOES HERE.
      DO 8 I=1,INODYO
      L=NODYO(I)
      C2=ABS(ZR(L)-ZCOR)
      IF(C2.LT.EPSI) GO TO 10
      GO TO 8
10     CX=ABS(XR(L)-CRACK)

```

```

      IF(CX.LT.EPSI) GO TO 9
8      CONTINUE
9      IF(NTYP.EQ.0) GO TO 12
      JSUM=0
      DO 90 IO=1,INODYO
      NS=NODYO(IO)
      C1=ABS(ZR(NS)-ZCOR)
      IF(C1.LT.EPSI) GO TO 130
      GO TO 90
130     JSUM=JSUM+1
      LINE(JSUM)=NS
90      CONTINUE
      ITNODX=JSUM
      DIST=YOUNG
      DO 91 IP=1,ITNODX
      LM=LINE(IP)
      CSD=XR(L)-XR(LM)
      IF(CSD.LE.0.0) GO TO 91
      IF(CSD.GT.EPSI.AND.CSD.LT.DIST) DIST=CSD
91      CONTINUE
      ICAM=0
      MODF=LM
      DO 92 LU=1,INODYO
      C10=ABS(XR(LU)-XR(MODF))
      IF(C10.LT.EPSI) GO TO 96
      GO TO 92
96      ICAM=ICAM+1
92      LBFOR(ICAM)=LU
      LTOTB=ICAM
      DO 94 JO=1,LTOTB
      LA=LBFOR(JO)
      STR=VOLD(LA)
      WRITE(6,15)LA,STR,PT
15      FORMAT(2X,"CRACK-TIP NODE",I4,"HAD",E11.4,"CTOD AT"
1      ,E11.4)
      KLU=0
      IF(STR.GE.(0.98*SCRIT))KLU=1
      IF(KLU.EQ.1) NBREAK=NBREAK+1
      IF(KLU.EQ.0) GO TO 997
94      CONTINUE
12      II=0
      DO 13 JJ=1,INODYO
      LL=NODYO(JJ)
      C3=XR(LL)
      C1=XR(L)
      C5=YR(L)
      C6=YR(LL)
      C7=ABS(C5-C6)
      C4=ABS(C3-C1)
      IF(C4.LT.EPSI.AND.C7.LT.EPSI) GO TO 155
      GO TO 13
155     II=II+1
      LOCAT(II)=LL
13      CONTINUE
      ICUT=II
      DO 16 JI=1,ICUT
      LC=LOCAT(JI)
      NV=3*LC-1
      MPTAB(NV)=0
      KLU=1
      IF(NBREAK.EQ.5) KLU=3
      ALP=-SK2
      Z(NV)=1.
      NC=NV
      IFAC=3
      CALL SYMBAN(LNSTIF,NDOF,MB,MSUM,AA,1,BB,IFAC,T1,IERR,

```

```

1 ALP,Z,T2,T3,T4,NC)
WRITE(6,100)LC,PT
100 . FORMAT(/,2X,"NODE",I4,"BROKE AT",F8.3/)
16 . CONTINUE
. CMIN=1.0E+10
. DO 36 JS=1,ICUT
. LO=LOCAT(JS)
36 . FORCE(JS)=-SK2*VOLD(LO)
. DO 60 IJ=1,INODYO
. LT=NODYO(IJ)
. C5=XR(LT)-XR(L)
60 . IF(C5.GT.EPSI.AND.C5.LT.CMIN) CMIN=C5
. CRACK=CRACK+CMIN
997 . RETURN
. END
SUBROUTINE SYMBAN(MAXN,H,M,MSUM,A,NRHS,B,IFAC,P,IERR,ALP,Z,W,D,T4, SYMBAN 2
C NC)
C SYMBAN 3
C SOLVE MATRIX EQUATION AX=B WHERE A IS SYMMETRIC POSITIVE DEFINITE SYMBAN 4
C AND B IS A MATRIX OF CONSTANT VECTORS. SYMBAN 5
C SYMMETRY AND BAND WIDTH OF MATRIX A IS ACCOUNTED FOR BY STORING A(I SYMBAN 6
C IN LOWER TRIANGULAR MATRIX (INCLUDING DIAGONAL) BY ROWS SYMBAN 7
C SYMBAN 8
C MAXN - MAXIMUM DIMENSION OF MATRIX A SYMBAN 9
C SYMBAN 10
C N - NUMBER OF ROWS OF MATRIX A SYMBAN 11
C SYMBAN 12
C M(I) - BAND WIDTHS (LARGEST NUMBER OF NON-ZERO COLUMNS TO THE LEFT SYMBAN 13
C AND INCLUDING THE DIAGONAL OF ROW I IN MATRIX A SYMBAN 14
C M(I) MUST BE GREATER THAN OR EQUAL TO 2 WITH M(1)=1 SYMBAN 15
C SYMBAN 16
C MSUM(I) - AN ARRAY COMPUTED IN SYMBAN SYMBAN 17
C SYMBAN 18
C T4(MXBND) WORKING STORAGE OF LENGH MAX BAND WIDTH.
C
C MXBND = MAX BAND WIDTH. T4 IS DIMENSIONED ONLY IN MAIN.
C
C NRHS - NUMBER OF RIGHT-HAND SIDES OF COLUMN VECTOR B SYMBAN 19
C SYMBAN 20
C IFAC - INPUT INTEGER SPECIFYING WHETHER OR NOT A CHOLESKY SYMBAN 21
C DECOMPOSITION OF MATRIX A IS TO BE COMPUTED SYMBAN 22
C WHERE A = L*D*L**T SYMBAN 23
C SYMBAN 24
C = 0 CHOLESKY DECOMPOSITION IS COMPUTED. IFAC SET TO 1. SYMBAN 25
C SYMBAN 26
C = 1 THE CHOLESKY DECOMPOSED FORM OF MATRIX A IS INPUT. SYMBAN 27
C SOLUTION IS RETURNED IN B. SYMBAN 28
C SYMBAN 29
C = 2 THE CHOLESKY DECOMPOSITION OF MATRIX A IS MODIFIED. SYMBAN 30
C MODIFICATION IS OF THE FORM A = A + ALP*Z*Z**T. SYMBAN 31
C NC IS THE FIRST NONZERO ROW IN COLUMN VECTOR Z. SYMBAN 32
C IFAC IS SET TO 1 AND Z SET TO 0 UPON RETURN. SYMBAN 33
C SOLUTION IS RETURNED IN B. SYMBAN 34
C SYMBAN 35
C = 3 ONLY THE CHOLESKY DECOMPOSITION OF MATRIX A IS MODIFIED SYMBAN 36
C IFAC IS SET TO 1 AND Z SET TO 0 UPON RETURN. SYMBAN 37
C SYMBAN 38
C DIMENSION A(MAXN),B(N,NRHS),P(N),W(N),D(N),Z(N),M(N),MSUM(N),T4(1) SYMBAN 39
C IF (IFAC.GT.0) GO TO 11 NEW 180
C LL=1 SYMBAN 41
C DO 10 I=1,N SYMBAN 42
C MSUM(I)=LL NEW 181
C LL=LL+M(I) SYMBAN 44
10 CONTINUE SYMBAN 45
11 CONTINUE NEW 182
IERR=0

```



CALL QSOLV(A,MSUM,H,B,P,N,IFAC,D,W,Z,T4,ALP,NC,IERR)	NEW	183
RETURN	SYMBAN	48
END	SYMBAN	49
· SUBROUTINE QSOLV(AR,IB,IL,B,DI,N,NFACT,D,W,Z,T,ALP,NC,IERR)	NEW	
· DIMENSION AR(1),IB(1),IL(1),B(1),DI(1),D(1),W(1),Z(1)	NEW	
· DIMENSION T(1)	NEW	
· DESCRIPTOR AV,BV	QSOLV	
· IF(NFACT.GE.2) GO TO 300	NEW	
· IF (NFACT.NE.0) GO TO 160	QSOLV	
C FACTOR	QSOLV	
DO 100 I=1,N	QSOLV	
ICI=I-IL(I)+1	QSOLV	
T(1;IL(I))=AR(IB(I);IL(I))	QSOLV	
N1=IB(I)+I-ICI	QSOLV	
AR(N1)=-1	QSOLV	
DO 100 J=ICI,I	QSOLV	
ICJ=J-IL(J)+1	QSOLV	
KS=MAXO(ICI,ICJ)	QSOLV	
N1=KS-ICI+1	QSOLV	
N2=J-KS+1	QSOLV	
ASSIGN AV,T(N1;N2)	QSOLV	
N1=IB(J)+KS-ICJ	QSOLV	
ASSIGN BV,AR(N1;N2)	QSOLV	
C=Q8SDOT(AV,BV)	QSOLV	
N1=J-ICI+1	QSOLV	
T(N1)=-C	QSOLV	
IF (J.EQ.I) GO TO 110	QSOLV	
N2=IB(I)+J-ICI	QSOLV	
AR(N2)=T(N1)*DI(J)	QSOLV	
GO TO 100	QSOLV	
110 CONTINUE	QSOLV	
IF(T(N1).LE.0.0) GOTO 999		
DI(I)=1/T(N1)	QSOLV	
D(I)=T(N1)		
100 CONTINUE	QSOLV	
C FORWARD SUBSTITUTION	QSOLV	
160 CONTINUE	QSOLV	
DO 200 I=1,N	QSOLV	
ICI=I-IL(I)+1	QSOLV	
ASSIGN AV,AR(IB(I);IL(I))	QSOLV	
ASSIGN BV,B(ICI;IL(I))	QSOLV	
C=Q8SDOT(AV,BV)	QSOLV	
B(I)=-C	QSOLV	
200 CONTINUE	QSOLV	
C DIAGONAL	QSOLV	
B(1;N)=B(1;N)*DI(1;N)	QSOLV	
C BACKWARD SUBSTITUTION	QSOLV	
NM1=N-1	QSOLV	
DO 400 II=1,NM1	QSOLV	
I=N-II+1	QSOLV	
ICI=I-IL(I)+1	QSOLV	
IF (ICI.GE.I) GO TO 400	QSOLV	
B(ICI;I-ICI)=B(ICI;I-ICI)-AR(IB(I);I-ICI)*B(I)	QSOLV	
400 CONTINUE	QSOLV	
C SOLUTION IS NOW IN B	QSOLV	
C D(1;N)=1.0/DI(1;N)	NEW	
RETURN	QSOLV	
C	NEW	
C***** NFACT = 2 OR 3 *****	NEW	
C	NEW	
300 CONTINUE	NEW	
DO 310 J=NC,N	NEW	
D(J)=D(J)+ALP*Z(J)*Z(J)	NEW	
BETA=ALP*Z(J)/D(J)	NEW	
ALP=ALP/(D(J)*DI(J))	NEW	
DI(J)=1.0/D(J)	NEW	



## REFERENCES

1. O. C. Zienkiewicz, "The Finite Element Method in Engineering Science," McGraw-Hill Book Company, New York, 1971.
2. N. Levy and P. V. Marcal, "Three-Dimensional Elastic-Plastic Stress and Strain Analysis for Fracture Mechanics," Division of Engineering, Brown University, HSST Technical Report No. 12, Dec. 1970.
3. C. S. Desai and J. F. Abel, "Introduction to the Finite Element Method, von Nostrand Reinhold Company, New York, 1972.
4. G. G. Pope, "A Discrete Element Method for Analysis of Plane Elasto-Plastic Strain Problems," R. A. F. Farnborough T. R. 65028 (1965).
5. T. L. Swedlow, M. L. Williams and W. M. Yang, "Elasto-Plastic Stresses in Cracked Plates," Calcit, Report SM, 65-19, California Institute of Technology (1965).
6. P. V. Marcal and I. P. King, "Elastic-Plastic Analysis of Two Dimensional Stress Systems by the Finite Method," Int. J. Mech. Sc., 9, 143-155 (1967).
7. S. F. Reyes and D. U. Deere, "Elasto-Plastic Analysis of Underground Openings by the Finite Element Method," Proc. 1st Ins. Congr. Rock Mechanics, 11, 477-86, Lisbon (1966).
8. E. P. Popov, M. Khojasteh-Bakht and S. Yaghmai, "Bending of Circular Plates of Hardening Material," Intern. J. Sol. Struc., 3, 975-988 (1967).